UNIVERSITÉ DE MONTRÉAL

SCHEDULING MATERIAL HANDLING IN CROSS-DOCKING TERMINALS

MOHAMMAD YOUSEF MAKNOON
DÉPARTEMENT DE MATHÉMATIQUES ET DE GÉNIE INDUSTREL
ÉCOLE POLYTECHNIQUE DE MONTRÉAL

THÈSE PRÉSENTÉE EN VUE DE L'OBTENTION
DU DIPLÔME DE PHILOSOPHIÆ DOCTOR
(GÉNIE INDUSTRIEL)
JUIN 2013

UNIVERSITÉ DE MONTRÉAL

ÉCOLE POLYTECHNIQUE DE MONTRÉAL

Cette thèse intitulée :

SCHEDULING MATERIAL HANDLING IN CROSS-DOCKING TERMINALS

présentée par : MAKNOON Mohammad Yousef
en vue de l'obtention du diplôme de : Philosophiæ Doctor
a été dûment acceptée par le jury d'examen constitué de :

M. ROUSSEAU Louis-Martin, Ph.D., président
M. BAPTISTE Pierre, ing., Doct., membre et directeur de recherche
M. SOUMIS François, Ph.D, membre et codirecteur
M. LAPORTE Gilbert, Ph.D., membre
Mme KAZEMI ZANJANI Masoumeh, Ph.D., membre

To my dear parents
and my beloved Shadi
for their unconditional
love and support

ACKNOWLEDGMENT

I wish to express my sincere appreciation to many individuals and organizations who made this research possible.

Special acknowledgment is made to Professor Pierre Baptiste for directing the thesis with his well understanding, encouragement and guidance during the research.

Grateful acknowledgment is made to Professor François Soumis for providing invaluable suggestions during my studies.

Finally, I wish to extend my deepest appreciation and gratitude to my parents, Reza and Fariba, my best friend, Shadi, my parents-in-law, Reza and Nooshin, and my sister and brother, Maryam and Saeed, not only for their encouragement, but also for their help and support in making this research possible.

This work has been partially supported by Fonds québécois de la recherche sur la nature et les technologies.

RÉSUMÉ

La manutention au sein des plateformes de distribution est un problème d'ordonnancement. Le transport interne des produits doit en effet être synchronisé avec les arrivées et les départs des camions. Ce problème se retrouve dans toutes les plateformes de distribution où la manipulation des produits est effectuée manuellement par l'opérateur.

Dans cette thèse, nous investiguons ce problème d'ordonnancement dans les plateformes de distribution. Nous mettons en relief les différentes facettes de ce problème et proposons une classification de ses différents sous problèmes. De manière générale, l'objectif est d'éviter les doubles manipulations (déplacer un produit d'un camion vers le stock, puis du stock vers un camion) qui doublent les coûts sans valeur ajoutée. Il faut minimiser ces doubles manipulations en orchestrant les transferts internes et la séquence de chargement/déchargement des camions.

Dans une première partie, nous analysons la structure du problème avec un modèle simplifié n'ayant qu'un quai de réception et un quai d'envois. Nous formalisons les décisions de manipulation interne et développons un algorithme optimal pour déterminer le meilleur plan de transfert de produits lorsque la séquence des camions est connue. Cet algorithme est utilisé comme fonction d'évaluation dans une recherche stochastique pour minimiser les doubles manipulations et optimisant les séquences de chargement/déchargement. Nous présentons ensuite un modèle de programmation linéaire en nombres entiers du problème général (ordonnancement des arrivées et départs de camions et transfert interne des produits). Nous proposons un algorithme de séparation et d'évaluation permettant une résolution efficace du problème. Nous proposons des structures de dominance et quelques inégalités valides permettant d'améliorer les performances de l'algorithme. Cette approche nous permet de résoudre à l'optimum en un temps raisonnable de très gros problèmes.

Dans une seconde partie, nous étendons ces modèles au problème général avec plusieurs quais. Nous nous intéressons d'abord au terminal de type satellite où l'ordonnancement des camions d'entrée est connu. Ces plateformes opèrent en deux mouvements différents : l'ordonnancement et chargement pour le transport de nuit et celui pour les livraisons matinales. Nous donnons une représentation mathématique qui permet de résoudre les problèmes de petite taille. Pour ceux de plus grandes ampleurs, nous utilisons une heuristique. Les résultats numériques montrent la validité de cette approche.

Finalement, nous généralisons le type de plateforme (les séquences d'arrivée et de départ sont à déterminer) et développons un nouveau modèle d'ordonnancement plus compact. Nous utilisons pour les grandes instances une recherche par voisinage. Nous mettons en place des voisinages originaux adaptés à ce type d'ordonnancement.

Mots clés : Transfert de produits, ordonnancement, plateforme de transbordement, recherche stochastique, programmation à nombres entiers, heuristiques, recherche du plus proche voisin.

ABSTRACT

Material handling in cross-dock is a relevant class of scheduling problems in distribution centers in which inner transhipment decisions need to be considered in addition to the processing order of trucks. The problem has applications in distribution centers where operators manually perform internal transhipment.

In this dissertation, we investigate the problem of material handling inside cross-docking terminals. The main component of the problem is presented, followed by a classification scheme to express its diversity. Moreover, double handling identifies the main source of deficiencies in transferring operations. The objective is to synchronize the trucks' loading and unloading sequences with internal transferring decisions to minimize excessive product displacement inside the terminal.

First, the problem is studied for a conceptual model of the platform with single receiving and shipping doors. We formalize decisions on internal transhipment and develop an algorithm to determine the best transferring plan with restricted orders on processing trucks. This algorithm is employed as an evaluation function in a stochastic search framework to ameliorate the order of processing trucks and reduce the cost of double handling. Then, a mixed integer linear programming formulation of the general problem is introduced. The proposed model determines the joint schedule between processing order of trucks at inbound and outbound doors with an internal transhipment plan. A path branching algorithm is proposed. We present several structural properties and some valid inequalities to enhance the performance of the algorithm. This method could solve fairly large instances within a reasonable time.

Second, we extend the developed models and approaches to schedule material handling process for a real platform with multiple doors. In the first installment, we focus on the satellite cross-docks that have limitations on the processing order of trucks at inbound door. These platforms operate in two separate shifts: consolidating pickup freight for overnight shipments and processing received products for early morning deliveries. A mathematical formulation of the problem is presented that can solve small instances with commercial software. In addition, a sequential priority-based heuristic is introduced to tackle the large problems. Numerical results depict the stability of this approach.

Finally, in the second instalment, we study the general model with no restriction on the arrival and departure pattern of trucks and formulate a new mathematical model. This model has considerably fewer variables and constraints than the previous one. Moreover, a variable neighborhood search heuristic is developed to tackle real life problems. This method consists of several operators incorporated in a search subroutine to find local optima and a perturbation operator to alter it. The developed method is adopted for three scenarios concerning limitations imposed by the network schedule. The analyzes demonstrate economical savings in the cost of material handling.

Keywords: Material handling; scheduling; cross-dock; stochastic search; Integer programming; heuristic; variable neighborhood search.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

INTRODUCTION

A cross-dock is a platform in a logistics network that receives freight from a supplier for several destinations and combines them with other suppliers' products for a common final delivery point (Kinnear (1997)). An economical advantage of cross-docking in a logistic network is its ability to consolidate different products to realize a full truck-load shipment without being dependent on platform inventory. In other words, the function of cross-docking is more about product coordination than product storage (Waller *et al.* (2006)). In addition, cross-docking has other benefits such as reducing order cycle time. These advantages are achievable with some considerations : operative transshipment of products in the logistic network, efficient usage of vehicle capacity (full truckload has priority), and a suitable cross-dock scheduling system to perform its duties at minimum expense (Apte et Viswanathan (2000)). The focus of this dissertation is on reducing operational costs by applying a convenient scheduling system for cross-dock platforms.

The cross-dock scheduling problem deals with organizing the process of material handling at the platform. These problems aim to organize the internal transshipment, which leads to reduced operational time and expense. A suitable model has to answer three interrelated questions : (1) when to process trucks (2) where to assign trucks and (3) how to transfer products.

The first question determines the arriving and leaving order of trucks. Sometimes platforms have to respect the restricted truck schedule that is imposed by the network (e.g., cross-docks in the postal service). The second and third questions deal with the problem of material handling. The second question provides proper dock allocation to minimize the average transferring distance within doors that causes a reduction in transfer time. The third question deals with the problem of double handling. A critical performance indicator for internal transshipment is the handling rate, which is equal to the amount of product transfer per shift with a given transporter and labor force. Since all arriving items must be transferred, and the amount of processing freight is known, the aforementioned performance measure would be optimized if one minimizes the number of total movement operations required to transfer all of the loads.

In this thesis, we give priority to the aforementioned indicator and propose scheduling models to minimize the cost of double handling. The scheduling model synchronizes the processing order of trucks with the internal transshipment decisions that minimize the excessive transferring procedure inside the terminal. In the following paragraphs, we outline the contributions of each chapter.

In chapter 1, we provide a review of truck scheduling problems in cross-docking terminals. We present a taxonomy of scheduling problems and survey previous research studied in cross-docking operations.

In chapter 2, we formalize the problem of double handling for a platform setting consisting of a single receiving and shipping door. We propose a dynamic programming model to optimize internal transshipment for the case of a known arrival and departure sequence of trucks. Then, we develop a stochastic search framework using the dynamic programming algorithm as an optimizer to reduce excessive handling of tasks by coordinating the processing periods of trucks.

In chapter 3, we provide, for the same platform setting, an exact resolution approach. A mixed integer linear programming model is formulated. We then introduce some families of valid inequalities that tighten the linear relaxation. Several structural properties are proposed. We embedded these properties in a branch and bound algorithm to find the optimal solution. The computational results depict that this algorithm can solve relatively large problems in a reasonable time.

A real cross-dock setting has many receiving and shipping doors. In the previous models, the processing order of trucks is equivalent with the time that a truck stays at the platform. However, this case is not valid for a platform with multiple doors. That means trucks are processed based on their arrival and departing orders.

Chapter 4 studies the scheduling problem in satellite cross-docks, which are responsible for local deliveries. These terminals operate in two separate shifts : consolidating pickup freight for overnight shipments and processing received products for early morning delivery. In these platforms, the arriving order of trucks is known. Therefore, a scheduling model simultaneously determines the leaving order of inbound trucks, processing period of outbound trucks and internal transferring decisions. We adopt the mathematical model formulate in chapter 4 for this type of platform. Moreover, we present a sequential priority-based heuristic that can be applied to real word problems. The computational results demonstrate efficient performance of the heuristic approach.

Finally, in chapter 5, we study a general scheduling case in which the platform has complete flexibility in assigning trucks. We present a different mathematical representation. The formulation has considerably fewer variables and constraints compared to the previous one. Also, we provide a variable neighborhood search heuristic. We introduce various search operators to look for the local optima in a decent subroutine and a perturbation operator to escape from the local optima. The heuristic demonstrates good behavior in terms of solutions for our test instances. In addition, we perform a comparative study between different scheduling scenarios and demonstrate savings in the cost of double handling.

CHAPTER 1

THE SCHEDULING PROBLEM AT A CROSS-DOCK TERMINAL : CLASSIFICATION AND LITERATURE REVIEW

Cross-docking is a logistic strategy used to coordinate inbound and outbound shipments that avoids having to store products between supplier and consumer . In the distribution process, incoming trucks deliver products to a cross-dock in order to be sorted and bundled as a single shipment ; afterwards, they are transferred to departing truck to be dispatched to their final destination. Moreover, cross-docking eliminates any additional storage between the supplier and customer, which results in a reduction in inventory holding cost (Apte et Viswanathan (2000)).

However, the use of cross-docks implies some obstacles. First, it increases the total transfer time, as additional stops at the platform need product displacement. Second, the consolidation process introduces additional variables to the main transportation problem and increases the fixed cost of staff and resources that may result in a decrease in the efficiency of the delivery process.

1.1 Elements impacting the cost of cross-dock

In order to make the process of consolidation within a cross-dock as efficient as possible, there is a range of decisions that need to be solved a priori. These decisions are divided into three categories based on their time frame and longevity effects (Stephan et Boysen (2011)).

1. **Network considerations**
   Using a cross-dock in a logistic network requires additional considerations in network design and implementation. A cross-dock should be located in a position that provides economies in transportation cost. In other words, particular attention must be paid to the flow of products that passes through the platform when determining platform location (Van Belle *et al.* (2012)). Moreover, the network schedule should permit consolidation at the cross-dock.
2. **Platform shape and layout**
   The shape and layout of the cross-dock has a considerable effect on its performance. Simply put, a decent platform size can significantly improve the efficiencies of its operations. The standard platform shape is a rectangular I-shape. The narrow I-shape terminal results in an increment in internal congestion. For larger platforms, in terms

of doors, other alternatives such as, L, X, T and H shapes are recommended. A fruitful discussion on the most suitable platform shapes has been introduced by Bartholdi et Gue (2004).

3. **Scheduling systems**

   Scheduling systems can be expressed based on the scarce resources of the terminal. According to Van Belle *et al.* (2012) there are two types of scheduling problems at the cross-docking platform.

   (a) **Dock door assignment**

   In dock door assignment, the platform has adequate dock to process all inbound trucks. Therefore, the scheduling decision is to find the proper allocation of trucks at the terminal, which minimizes the average transfer distance within doors.

   (b) **Truck sequencing problems**

   In dock door assignment, we assume that the amount of docks is the same as the amount of trucks. Therefore, each truck can be assigned to a different door and we do not need to consider the time factor. However, if this assumption is violated, dock doors are considered limited resources that have to be scheduled over time (Van Belle *et al.* (2012)). This introduces a new series of problems, which are called truck-sequencing problems.

   These problems suggest several questions about when a truck has to be processed or how the products should be transferred. Depending on the nature of the industry and the platform specifications, the scheduling problems differ.

To conclude, cross-docking requires proper decisions in different levels. In this thesis, we study truck scheduling systems. More specifically, our focus is on truck sequencing problems to synchronize the platform's internal operations so platform cost is kept to a minimum. In the following section, we present the classification of these problems.

1.2 Truck sequencing problem

In general, cross-docking platforms utilize a dock management system to coordinate truck allocation. This is an information system that is responsible for queuing and allocating trucks at the platform. At the entrance to the terminal, the incoming trucks are registered and assigned to a parking position. Once a platform dock is available, the dock management system selects a truck from the queue and allocates it to the dock for processing.

At the platform, the assigned truck is unloaded and the products are scanned and marshaled to their final destinations. Inside the terminal, based on the shape and the size of products, one of the automatic or manual handling systems is applied for internal transhipment.

For small and medium size packages, an automatic handling system is a suitable option. It is a set of highly automated conveyors and sortation systems that connect inbound to outbound doors. The advantage of a conveyor system is its relatively low transferring cost associated with products. Therefore, the objective of the scheduling model is to synchronize products' loading and unloading periods subject to internal transferring distance and congestions.

In contrast, if the products are oddly shaped, their weight and dimensions vary considerably. Implementing an automatic system becomes a challenging and costly task. Consequently, the companies prefer to apply manual handling devices such as forklifts or pallet jacks. The advantage of this system is its flexibility with transferring all types of goods. However, the associated transfer cost for each product is considerably higher than the automated system, as the transporters and operators are directly engaged with each product displacement. Here, in addition to elements described for the automated system, the manner of displacing products affects the operational cost. That means any excessive movement of products is costly and should be avoided.

To conclude, regardless of the type of internal transhipment method, three elements should be considered in truck sequencing problems : 1) the processing periods of trucks at the platform, 2) truck-dock allocation, which minimizes total transfer distance, 3) internal transhipment decisions to avoid excessive displacement of products. Figure 2.1 illustrates these elements.

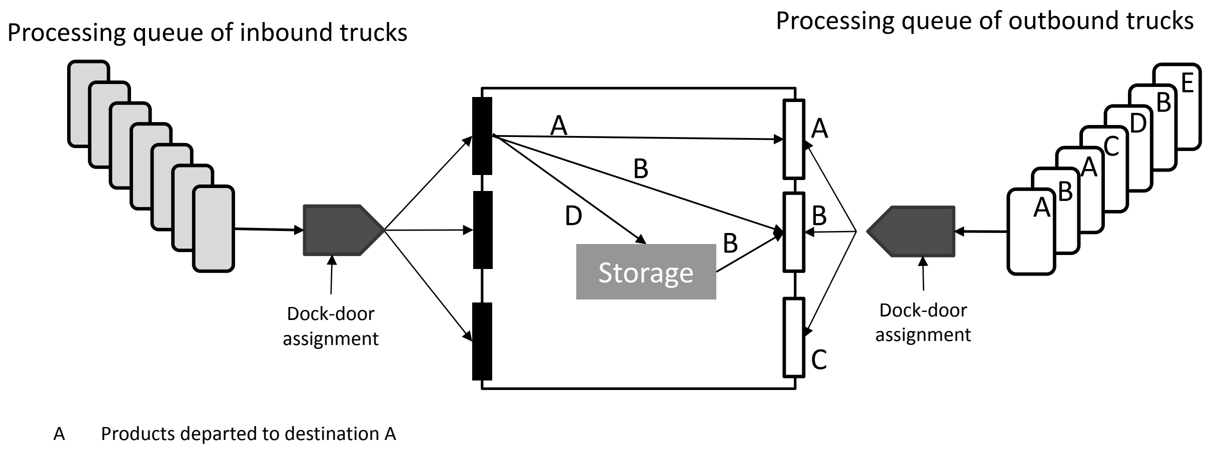
Processing queue of inbound trucks
Processing queue of outbound trucks
A
B
D
B
Storage
A
B
C
E
B
D
C
A
B
A
Dock-door assignment
Dock-door assignment
A Products departed to destination A

Figure 1.1 A Schema of cross-docking operation

Based on the aforementioned explanations, the characteristics that scheduling depends on are grouped in two main categories. Some of these characteristics are adopted from Van Belle *et al.* (2012).

1. **Physical attributes**

   (a) The size of the platform

   The research and studies in truck sequencing problems consider two platform settings. In the first setting, it is assumed that the platform has a single receiving and single shipping door, while in the second setting, the platform studied has multiple receiving and shipping doors.

   (b) Internal storage

   Although the goal of cross-docking is to eliminate storage, in reality most of the platforms use a small temporary storage area that helps them synchronize their operations. However, in some industries there are cases where storing is problematic (e.g., refrigerated products). In these platforms, internal storage is forbidden.

   (c) Material handling mode

   The technology and methods that are applied within a platform to transfer products could be manual or automated. In the manual transferring system, forklifts or pallet jacks are used to carry goods. This process is labor intensive and time consuming, which makes scheduling problems more complicated to solve. However, in automated systems, more advanced conveyor systems are used to relocate products. The main objective of the scheduling problem in automated systems is to minimize the duration of loading and unloading operations.

2. **Operational attributes**

   (a) Restriction on truck availability

   The arrival time of the inbound trucks determines the arrival time of products, which has a deep impact on the congestion of the cross-dock and scheduling worker and resources plans. Timings are important in planning the process of loading and unloading products.

   Some researchers have assumed all trucks are available at the beginning of the planning horizon. The examples of these models are the satellite cross-dock, which processes freight in two periods : early morning or late afternoon. There are also cases in which the arrival pattern of the trucks is distributed throughout the day.

   (b) Restrictions on processing trucks

   The restriction on the processing period of trucks is defined based on the flexibilities that the platform has on scheduling inbound and outbound trucks. For example, in parcel delivery industries, the cross-dock faces restrictions on the arrival and departure time of trucks (There is a penalty for a late shipment). In contrast, if the preference of the logistic network is on product consolidation, then the platform has more flexibility with scheduling the processing period of trucks.

(c) Pre-emption

Pre-emption allows the loading or unloading process of a given truck to be temporary, interrupted by replacing the one in process with another truck in the queue.

(d) Consolidation

At the platform, there are two types of consolidation. In the first type, the inbound products are distinguished by their type and each outbound truck requires a certain number of products. In the second type, all inbound products are distinguished based on their final destination and the consolidation process serves to bundle products for the same destination.

1.3 Literature review

In this section, we review papers that have tackled the problem of truck sequencing in cross-docks. The studies are presented in two groups. The first group represents research that uses an automatic system for internal transhipment. The second group contains ones that employ the manual transhipment method. A summary of these papers is presented in Table 1.1.

1.3.1 Truck sequencing – with automated internal transhipment mode

These types of scheduling problems arise mostly in cross-dock operations in courier industries. The main scheduling task is to coordinate the arrival and departure time of the trucks. The objective function is to minimize operational cost, which is a function of processing time.

Chen et Lee (2009) have considered the two-machine cross-dock flow shop problem. The main objective is to plan sequences between inbound and outbound trucks while minimizing the time span from the beginning of the unloading the first inbound truck until the end of the loading of the last outbound vehicle. The authors demonstrate that this is an NP-hard problem. They have proposed a branch and bound algorithm that can solve instances up to 60 trucks in a reasonable time.

The extension of the aforementioned problem has been studied by Chen et Song (2009) for the two stage hybrid cross dock scheduling problem. In this case, numerous trucks can be loaded and unloaded at a given time. The travel time between inbound and outbound is not considered. An MIP model that applies several heuristics is suggested to tackle the problem.

There are cases in which the consolidation process is also considered ; that is, the product assignments from the inbound to outbound trucks have to be determined by the platform. Yu et Egbelu (2008) have studied such a case while minimizing the makespan. In their research, the travel time between receiving and shipping doors is fixed. They have introduced a mixed integer linear programming in parallel with some heuristic methods to solve large examples.

Vahdani et Zandieh (2010) have used the heuristic method of Yu et Egbelu (2008) as an initial solution to their five suggested meta-heuristic algorithms. These five methods include : genetic algorithm, Tabu search, simulated annealing, an electromagnetism-like algorithm and variable neighborhood search. The result shows an improvement while choosing the method of Yu et Egbelu (2008) as an initial solution. A similar method has also been suggested by Boloori Arabani *et al.* (2011), who represent five meta-heuristics to tackle this problem : a genetic algorithm, tabu search, particle swarm optimization, ant colony optimization and differential evolution.

Boysen *et al.* (2010) have also solved a similar problem by assuming that the time horizon is divided into different time intervals during which trucks can be fully loaded or unloaded. The problem is formulated as an integer programming model and it is shown that this problem is NP hard. They have used a decomposition approach in order to provide two sub-problems, which are solved iteratively and sub-optimally by using a heuristic approach based on dynamic programming. As soon as their defined stopping criterion is met, the global solution is obtained.

A new objective function has been determined by Arabani *et al.* (2010), in which the outbound trucks have a due date and the objective is to minimize the total earliness or tardiness of these trucks. They have suggested a genetic algorithm, particle swarm optimization and differential evolution to solve the problem. A similar problem has been considered by Vahdani et Zandieh (2010) in which temporary storage is not permitted. To directly transfer products from one door to another, the loading and unloading trucks can be stopped and postponed to a later time. The problem is formulated as an MIP model and two meta-heuristic methods including a generic and an electromagnetism-like algorithm are proposed to solve it. Moreover, the same problem has been tackled by Soltani et Sadjadi (2010) by using hybrid simulated annealing and hybrid variable neighborhood search methods.

Mcwilliams et al. (McWilliams *et al.* (2005) ;McWilliams *et al.* (2008) ;McWilliams (2009)) have studied a scheduling problem in parcel hub. The model includes planning the set of inbound trucks that are loaded with a batch of varied parcels to a set of shipping docks. The objective is to minimize the time span of transferring operations. They have suggested a simulation method that uses generic algorithm to lead the search scheme. They have also proposed a lookalike knapsack model to solve the problem. A genetic algorithm is used as heuristic method to deal with large sized problems. Moreover, they have shown that a proposed local search method and simulated annealing outperform genetic algorithm.

1.3.2 Truck sequencing – with manual internal transhipment mode

These groups of studies have a more comprehensive view of platform operations by including the internal transhipment in the objective of the scheduling model.

Larbi *et al.* (2011) have studied the scheduling of the outbound trucks in a cross-dock with a single receiving and a single shipping door. An arriving truck is unloaded. The products with the same destination are consolidated and marshaled to load at outbound trucks. The unloading procedure can be performed at any time and the outbound trucks are always available in this model. The target is to minimize the total cost, which includes the storage and the pre-emption costs. An algorithm is proposed that can solve the problem in polynomial time in a case where there is information about the inbound trucks. Unlike the first case, in the second part, it is assumed that no information is available about the content of inbound trucks. A probabilistic based heuristic algorithm is suggested in order to determine which outbound truck has to be loaded next. The last case is when partial information about the inbound truck arrivals is available. Two heuristic methods are suggested. In the first one, the approach for the full information case is adapted for a rolling horizon algorithm that is recalculated every time a new truck arrives. The second heuristic combines the algorithms for the full information and the no information cases. Results demonstrate that when no information is available, the cost increases.

Alpan *et al.* (2011b) have extended the problem to a case where a cross dock has multiple receiving and shipping doors. A dynamic programming approach is suggested to solve the problem.

Another case is investigated by Wang et Regan (2008) for scheduling outbound trucks. They have proposed a series of dispatching rules that make an online decision about the processing timing of trucks. They suggest two-time based algorithms, which consider the impact of the arrival of a new truck in the inbound doors on the total time of processing at the cross-dock (waiting time at the door and transferring products from one door to another). A simulation approach is studied to compare two algorithms under two different assumptions : first, the First-Come-First-Served rule (FCFS) and second, the look ahead policy. As a result, significant time has been saved by using their proposed time-based rules.

Boysen (2010) has dealt with a scheduling problem in which products are not permitted to be immediately stored. This zero inventory policy can be used for perishable products. This paper studied a cross-dock in the food industry. In this research, the travelling time between doors has been ignored. The objective is to reduce the processing time and tardiness of outbound trucks. A dynamic programming approach is proposed to solve this problem. For large size problems, simulated annealing is applied to find near optimal results in a reasonable amount of time.

Table 1.1 A classification of the reviewed papers [1]

Paper	**Physical attributes**			**Operational attributes**				**Elements of M.H.**		
	# Doors	**Storage**	**M.H. mode**	**Truck availability at the beginning of the planning horizon**	**Restriction on processing trucks**	**Pre-emption**	**Consolidation**	**(1)**	**(2)**	**(3)**
Chen et Lee (2009)	2	Yes	ns	Yes	No	No	N/A	Yes	No	No
Chen et Song (2009)	>2	Yes	ns	Yes	No	No	N/A	Yes	No	No
Yu et Egbelu (2008)	2	Yes	Automatic	Yes	No	No	Type	Yes	No	No
Vahdani et Zandieh (2010)	2	Yes	Automatic	Yes	No	No	Type	Yes	No	No
Boloori Arabani *et al.* (2011)	2	Yes	Automatic	Yes	No	No	Type	Yes	No	No
Boysen *et al.* (2010)	2	Yes	Automatic	Yes	No	No	Type	Yes	No	No
Arabani *et al.* (2010)	2	Yes	Automated	Yes	Outbound	No	Type	Yes	No	No
Vahdani et Zandieh (2010)	2	No	Automated	Yes	No	Yes	Type	Yes	No	No
Soltani et Sadjadi (2010)	2	No	Automated	Yes	No	Yes	Type	Yes	No	No
McWilliams *et al.* (2005)	>2	No	Automated	Yes	Outbound	No	Type	Yes	Yes	No
McWilliams *et al.* (2008)	>2	No	Automated	Yes	Outbound	No	Type	Yes	Yes	No
McWilliams (2009)	>2	No	Automated	Yes	Outbound	No	Type	Yes	Yes	No
Larbi *et al.* (2011)	2	Yes	Manual	No	No	Yes	Destintation	Yes	No	Yes
Alpan *et al.* (2011b)	>2	Yes	Manual	No	No	Yes	Destination	Yes	No	Yes
Wang et Regan (2008)	>2	No	Manual	No	No	No	Destination	Yes	No	No
Boysen (2010)	2	No	Manual	Yes	No	No	No	Yes	No	No

1 Some papremeters are adopted from Van Belle *et al.* (2012)

1.4 Characteristics of the platforms studied in this thesis

In this thesis, we study truck sequencing problems with explicit decisions on internal transhipment. The platform studied has the following physical and operational characteristics :

1. The model examined in chapter 2-3 has a single receiving and shipping door. In chapter 4 and 5, we study the scheduling model for a platform with multiple doors
2. Internal storage is permitted
3. The terminal use manual handling systems to transfer products
4. All of the trucks are available at the beginning of the planning horizon
5. We study the variety of cases regarding limitation on the arrival and departure order of trucks

6. Pre-emption is not allowed
7. We use product consolidation based on destinations
8. The model examined synchronizes the arrival and departure time of the trucks with the internal transferring plan.

CHAPTER 2

ARTICLE 1 : CROSS-DOCKING : INCREASING PLATFORM EFFICIENCY BY SEQUENCING INCOMING AND OUTGOING SEMI-TRAILERS

Chapter Information : A previous version of this chapter was published in International Journal of Logistics : Research and Applications MAKNOON, M. Y. et BAPTISTE, P. (2009). Cross-docking : increasing platform efficiency by sequencing incoming and outgoing semi-trailers. , 12, 249-261.

Abstract Cross-docking is a transhipment platform used to consolidate incoming products for outgoing destinations. Research in cross-docking mostly studies the cross-dock network (platform location) and the platform design (door locations). In this study we consider a platform's internal operation and we focus on the direct flow of products from receiving to shipping doors. This flow can be improved by the synchronization of incoming and outgoing semi-trailers. Two methods, dynamic programming and a heuristic approach integrated with a stochastic evolutionary algorithm are proposed as a resolution approach for this problem.

keyword Cross-docking, scheduling, sequencing, merchandise flow, heuristic method, stochastic evolutionary algorithm

2.1 Introduction

Cross-docking is a logistic facility between the producer and consumer with the function of product coordination rather than product storage. At the platform inbound door, the incoming products, which differ according to their sending destinations, are unloaded, broken down, processed and consolidated for reshipment at the outbound door. The consolidated products are either transferred directly to the loading semi-trailer (one pickup) or put into temporary storage (two pickups) for future reshipment.

The efficiency of a cross-docking platform depends highly on the generated products flow. Three factors are important in products flow : travel distance from inbound to outbound doors, congestion and their moving path. In this research we consider two moving paths for products : first, direct transfer from the receiving to the shipping door (in this case, a single manipulation is needed). Second, using intermediate storage for future reshipment (in this case two manipulations are needed).

In this paper, we focus on the importance of the movement of internal products and we propose a method to decrease them. In other words, our goal is to increase the number of products that transmit directly from the inbound to the outbound door by sequencing incoming and outgoing semi-trailers.

Platform efficiency can be defined as the ratio between the total numbers of direct transiting products to the total number of transiting products. To increase the efficiency, first we propose a dynamic programming algorithm to solve a restricted sub-problem in which both sequences are fixed. Then, this function is developed and integrated with an evolutionary algorithm as a resolution for the problem in which both sequences are variable. Moreover, a faster heuristic approach is presented and the results show that both algorithms have good performance. Finally, we conclude that sequencing incoming and outgoing semi-trailers notably increases the platform efficiency.

2.2 Cross-Docking operational problems

Cross-docking is defined as a transshipment platform that receives products from a supplier for several destinations and consolidates them with other suppliers' products for a common final delivery destination (Kinnear (1997)). An economical advantage of cross-docking in a logistic network is its ability to consolidate different products in order to have full truckloads without being dependent on a platform's inventory. In other words, the function of cross-docking is more product coordination than product storage (Waller *et al.* (2006)). In addition, Cross docking also has other advantages such as a reduction in the order cycle time. These benefits are achievable with some considerations such as : efficiently handling the flow of products, efficiently using semi-trailer capacity (a full semi-trailer load) ; and implementing a good scheduling system based on the proper information system (Apte et Viswanathan (2000)).

A cross-docking network has two major considerations : reducing global inventory and satisfying consumer demand at the right time. Zhang (1997) defined two types of networks : schedule-driven and load-driven. Schedule-driven is a cross-docking network for which delivery time is a priority rather than having fully loaded semi-trailers (e.g. the postal service network). On the other hand, in a load-driven network, a fully loaded semi-trailer is a priority. Ratliff *et al.* (2003) have studied load-driven networks for the automobile industry in order to obtain the number and location of each platform and the shipping flow between them. A similar study was done on the US postal service by Donaldson *et al.* (1998), examining a schedule-driven network.

Chen *et al.* (2006) have considered a more general network with a pickup and delivery time window to minimize the inventory level. Moreover, Lee *et al.* (2006) have studied vehicle routing and scheduling problems in a cross-docking network.

By studying cross-docking networks, the platform location, the number of semi-trailers and their arrival and departure times at each platform are determined. In this case, one problem that can arise is the ability of the platform to respect the determined schedule. Li *et al.* (2004) have studied the loading/unloading scheduling problem on a transshipment platform when each container has to be filled at an exact time.

Minimizing the platform processing time is studied in (Yu et Egbelu (2008), McWilliams *et al.* (2005)). McWilliams *et al.* (2005) have minimized the time interval between the first unloaded and the last loaded parcel in a freight consolidation terminal using a generic algorithm and simulation. Yu et Egbelu (2008) use heuristic methods to minimize the completion time by scheduling inbound and outbound semi-trailers in a platform with one receiving and one shipping door when the storage is located at the shipping dock.

Besides the cross-docking vehicle scheduling problems, platform revenue highly depends on its internal operations. In a simple operation, products that have arrived are sorted by the outgoing destination on the platform and then they are directed to an outgoing truck. In a more complex operation, products that have arrived blend with storage products and transfer to an outbound door ; the platform efficiency is highly dependent on the product movement inside the platform (Ackerman (1997)).

As presented in Figure 3.1, three issues arise for products that are transmitted inside the platform : first is the distance between loading and unloading doors. Second is the product flow congestion and third is the product's moving path (products that arrived are directly transferred to the shipping door (need one pickup operation) or moved to a storage area (need two pickup operations)). Tsui et Chang (1992) have formulated the dock assignment problem, simultaneously allocating both inbound and outbound doors to semi-trailers, as an integer programming model. They have proposed a microcomputer-based decision support tool to assign dock doors in a freight yard (Tsui et Chang (1990)).

Recently, Bottani *et al.* (2004) have considered a fuzzy multi-attribute dynamic time-based method to manage priorities for loading and unloading doors. In other studies, (Bartholdi et Gue (2000) ;Bartholdi et Gue (2004)) have minimized labor costs by developing models for travel cost within the docks and the congestion that occurs during consolidation. They stated that freight flow patterns are determined by platform layout, material handling systems, freight mix and scheduling.

As mentioned before, three issues are important for the movement of products inside the platform. In this study, our aim is to investigate the movement of products inside the

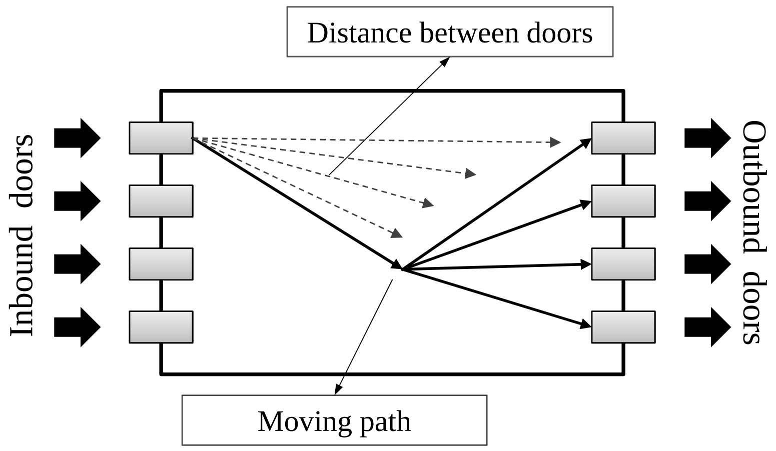

Figure 2.1 Product movement inside platform

platform in order to increase the efficiency of the platform. In other words, our goal is to increase the number of products that are transferring from inbound to outbound doors by sequencing incoming and outgoing semi-trailers.

Baptiste P (2007) have classified the semi-trailer sequencing problem in a cross-docking platform by considering flexibility of an incoming or outgoing sequence, number of semi-trailers for each destination and queuing model ; they state that in the case in which semi-trailers depart to one destination and incoming sequences are known, the problem is polynomial.

Overall, in this study, our objective is to minimize the movement of products inside the platform by sequencing incoming and outgoing semi-trailers, in the case that we have many semi-trailers for each destination with flexibility on semi-trailer sequences.

2.3 Model description and assumptions

In this research, our focus is on the number of movements for each product and we do not consider the moving distance and congestion. The impact of incoming and outgoing sequences of semi-trailers on the number of movement is unknown. To get a first idea of this impact, we have decided to study it on a restricted case (a single incoming door and a single outgoing door) in order to be able comparing our results to optimal ones (that can be obtained by enumeration in small case). Our goal is not to produce software that solves real problems, but to have an idea of the impact of all parameters : incoming sequence, outgoing sequence and loading and unloading policies.

As presented in Figure 2.2, at a cross-docking platform, stock keeping units (called in this paper "products") arrive at the platform ; the products are unloaded from the carrier (called in this paper the "semi-trailer"). If the outgoing semi-trailer departs to the products' final destination, the products move directly to the outbound door ; otherwise, they are transferred to temporary storage.

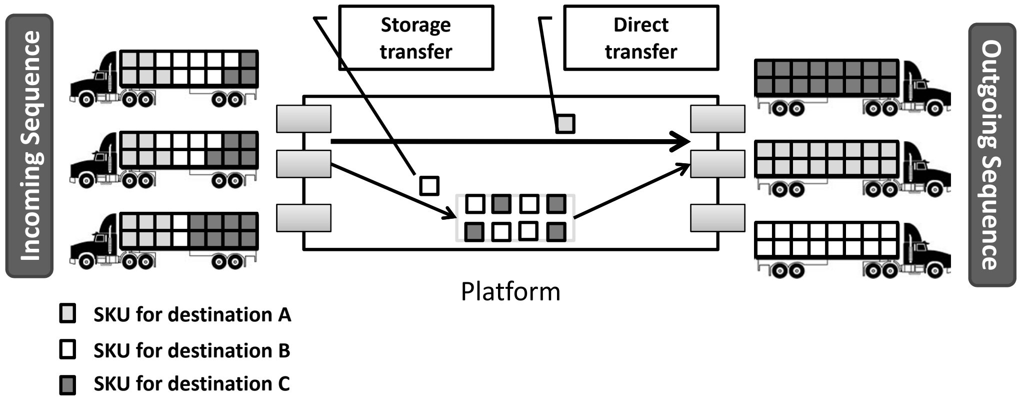

Figure 2.2 Sample cross docking platform

In this model, the following assumptions are considered :

– All semi-trailers use their maximum capacity (fully loaded).
– Each semi-trailer leaves the platform only when it is completely loaded or unloaded.
– The storage capacity is unlimited.
– Each outbound semi-trailer departs for only one destination.
– All incoming and outgoing semi-trailers are available at the start of the planning horizon (shift or day).
– Before products arrive at a platform they are distinguished by their outgoing destination.
– All transferring time inside the platform is constant and is not considered.
– The numbers and the capacities of incoming and outgoing semi-trailers are equal.
– There is no rule for unloading products from the semi-trailer.

Product movements in a platform are affected by four factors : the incoming sequence of semi-trailers, the outgoing sequence of semi-trailers, the unloading sequence of products from semi-trailers and the loading policy.

The first and second factors are the order of semi-trailers. For the third factor, suppose that each incoming semi-trailer contains items to be shipped to different destinations, "Items that can be shipped to the current outgoing semi-trailer must be unloaded first" ; this factor could be fix, due to technical constraints for the unloading operations. For example, if the products are unloaded according to the FIFO (First In First Out) or LIFO (Last In First Out) rules, the third factor is fix ; otherwise, it is variable.

For the fourth factor, suppose the following situation : an outgoing semi-trailer is positioned at the outbound door and some items are waiting in storage to be shipped to their destination. The manager can choose to reship those items or to wait until an incoming semi-trailer arrives with items that can be shipped directly to the destination. In this situation there are two extreme possible loading policies : either the storage products are systematically used to complete semi-trailers (less inventory), or storage items are shipped in the last semi-trailer going to their destination. It is shown that the optimal policy is a combination of these two extremes Maknoon (2007).

2.4 Resolution approaches

In this section two approaches are proposed. In the first approach, we first solve the restricted case in which both sequences are fixed. We use a dynamic programming algorithm to obtain optimal loading and unloading policies. Then, this algorithm is encapsulated as an evaluation function in a stochastic evolutionary algorithm for obtaining acceptable incoming and outgoing sequences.

For the second approach, we first solve with a heuristic the restricted case in which only the incoming sequence is fixed. This heuristic produces simultaneously an outgoing sequence and loading and unloading policies. Then, this heuristic is encapsulated as an evaluation function in an evolutionary algorithm that found a good incoming sequence.

Table 2.1 presents the general overview of two approaches ; in the following section each approach will be described in more detail.

Table 2.1 Resolution approaches

Approach	Sequencing incoming and outgoing algorithm (S.I.O.A)	Sequencing incoming and outgoing greedy algorithm (S.I.O.G.A)
Restricted case	Both sequences are fixed	Incoming sequence is fixed
Looking for	Loading/unloading policy	Outgoing sequence and loading/unloading policy
Method	Dynamic programming	Heuristic
Development	Integrated evolutionary algorithm to obtain good incoming and outgoing sequences	Integrate an evolutionary algorithm to obtain good incoming sequence

2.4.1 Sequencing incoming and outgoing algorithm (S.I.O.A)

First, we assume that both semi-trailer sequences are fixed and the objective is to order unloading products and their loading policy. To do that, we use a decision tree (sequential acyclic graph) in which each loading semi-trailer is considered as a state. At each state, each node is an alternative to fill the loaded semi-trailer. Moreover, in each node (option) the following information is stored :

- For each destination (d) Vector of variables indicates the possible directly transiting products (H_d)
- For each destination (d) Vector of variables indicates the number of products in temporary storage (S_d)
- Profit (P) (the total number of direct transiting products from the beginning state to the current option)
- Unloading semi-trailer order number (I)

As presented in Figure 2.3, the loading/unloading algorithm (L.U.A) starts with an initial option considering the data from the first unloading semi-trailer, then it selects the first loading semi-trailer and generates all of the possibilities to fill it. Each possibility is obtained by combining the storage products and products in a different unloading semi-trailer. After this procedure, for its current state, we have a list of all alternatives with different profits (number of direct transiting products). In the current state we are unable to select the best option but it is possible to filter them ; therefore, the domination function is used to filter the alternatives. This function consists of two rules that are described as follows :

The first rule states that, for two selected options, if for each destination directly transiting products for two options are equal. The option that has a higher profit and fewer unloading semi-trailers dominates the other. The second rule applies when two or more options have the same profit and the same unloading semi-trailer, the option with the higher summation of directly transiting products for all destinations dominates the others.

After applying the domination function, the algorithm considers the next loading semi-trailer ; this process continues until all loading semi-trailers are filled. At the end, the option in the last state having the highest profit is the optimal option and the path from the initial option to the selected final one is the optimal policy for loading and unloading products. For more algorithmic details about this method, readers should refer to Appendix 1 of this paper.

An illustrative example :

Suppose that there are five incoming and five outgoing semi-trailers with a capacity of ten products each. The outgoing semi-trailers depart to three destinations (two to destination A,

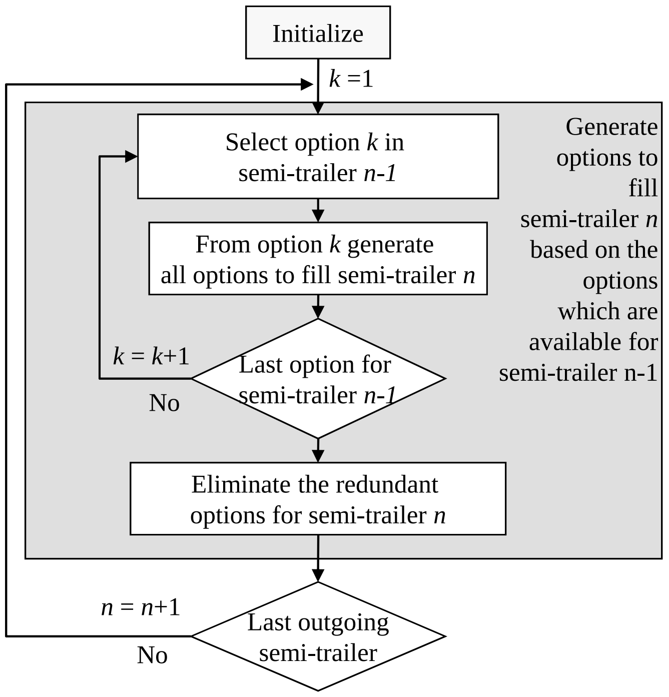
Initialize
k =1
Select option k in semi-trailer n-1
From option k generate all options to fill semi-trailer n
Last option for semi-trailer n-1
k = k+1
No
Eliminate the redundant options for semi-trailer n
Generate options to fill semi-trailer n based on the options which are available for semi-trailer n-1
Last outgoing semi-trailer
n = n+1
No

Figure 2.3 Loading and unloading algorithm (L.U.A)

two to destination B and one to destination C). The outgoing sequence is $A-B-A-B-C$, and the incoming sequence is $I-II-III-IV-V$. Table 2.2 presents the products in each semi-trailer ; for example, the first unloading semi-trailer has six products for destination A, three for B and one for C.

Table 2.2 Illustrative example

Incoming	Semi-trailer contents for each destination		
semi-trailer	A(products)	B(products)	C(products)
I	6	3	1
II	2	7	1
III	5	2	3
IV	3	4	3
V	4	4	2

The loading/unloading algorithm (L.U.A) is applied and the generated decision tree is presented in Figure 2.4.

The first option is an initialization. The algorithm reads the initial option and generates alternative 2 with the profit of 10 as a possible assignment for the first loading semi-trailer (destination A). From option 2, alternatives 3, 4 and 5 are generated for the second loading semi-trailer (destination B). This procedure continues until all outgoing semi-trailers are loaded. Options 3 and 5 have the same number of directly transiting products and the profit of option 3 is higher than option 5, but it did not satisfy the domination rule; on the other side, option 10 is dominated by option 9. In the fifth loading semi-trailer, option 12 has the highest profit (29), therefore, it is chosen as the best option and the path with options 1-2-3-6-9-12 is the optimal loading/unloading policy.

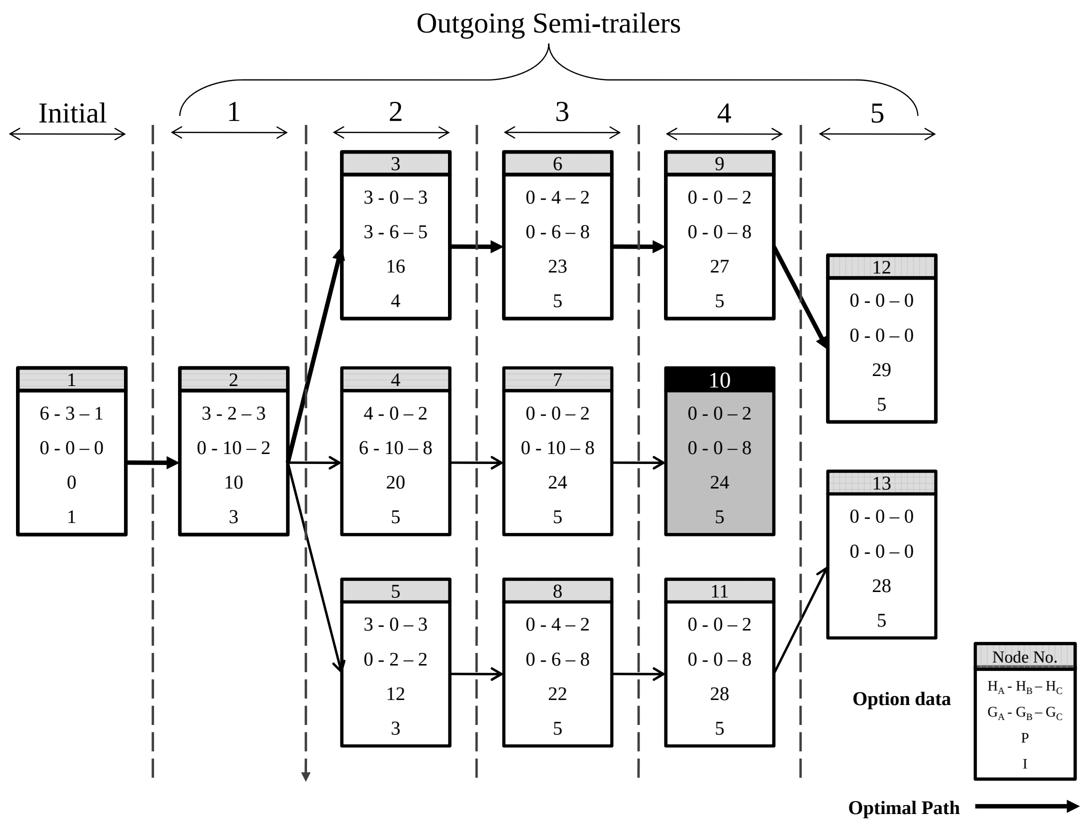

Figure 2.4 Illustrative example for L.U.A

Overall, when both sequences are fixed, the proposed dynamic programming algorithm reaches the optimal solution. Now we consider the case in which both sequences are variable and we are seeking to increase the profit by obtaining good sequences for incoming and outgoing semi-trailers. In this case, we integrate a stochastic evolutionary method with the proposed dynamic programming algorithm. As presented in Figure 2.5, the sequencing incoming and outgoing algorithm (S.I.O.A) starts with an initial solution by running L.U.A.

Then, the algorithm consecutively considers incoming and outgoing sequences and randomly swaps semi-trailer orders to improve the profit. This loop terminates when the profit does not improve after three successive iterations.

2.4.2 Sequencing incoming and outgoing greedy algorithm (S.I.O.G.A)

In S.I.O.G.A, first, we consider a restricted case in which the incoming sequence is fixed. In this situation we propose a heuristic method to simultaneously obtain a good outgoing sequence and loading and unloading policy. The heuristic is an iterative method and similar to L.U.A ; the state changes by loading semi-trailer. In each state the algorithm considers all the possibilities to fill the semi-trailer. Three factors are considered to construct the possibilities : different destinations, different unloading semi-trailers and different loading policies (selection between the storage products and direct transiting products). After all alternatives are generated, the algorithm selects the best one that has the highest efficiency (total number of direct transiting products to total number of products). Then the algorithm moves to the next state and starts from the best selected option in the previous state. This procedure ends when all loading semi-trailers are assigned to the destination.

Like the L.U.A, in the heuristic method each option has the following data :

- For each destination (d) Vector of variables indicates possible directly transiting products (H_d)
- For each destination (d) Vector of variables indicates the number of products in temporary storage (S_d)
- Profit (P) the total number of direct transiting products from the beginning state to the current option
- Storage (S) the total number of products move to storage from the beginning state to the current option
- Unloading semi-trailer order number (I)
- Loading path

Similar to the proposed dynamic programming approach, the stochastic evolutionary algorithm integrated with the heuristic method. However, in the heuristic method, the evolutionary algorithm is only used to swap the incoming sequence of semi-trailers. Figure 6 presents the algorithm flow chart.

Example

For our illustrative example, the cost obtained with the first algorithm (S.I.O.A) is 38 with the outgoing sequence $B-A-B-C-A$ and incoming sequence $II-I-V-IV-III$.

For the second algorithm (S.I.O.G.A), the outgoing sequence is $A - B - A - C - B$ with $I - V - II - III - IV$ as an incoming sequence and a cost of 37.

2.4.3 Experiments and discussion

In the previous sections, the resolution approaches for semi-trailer sequencing problems in transhipment platforms were studied. In this section, we test our proposed algorithms with generated sample data. The tests were performed with a Pentium 2.2 GHz computer.

To generate the data, we considered example problems with five, ten, twenty and forty incoming semi-trailers with the capacity of ten products for each semi-trailer. For the problem with five semi-trailers, three and five destinations were defined. For the rest of the problems, three, five and ten destinations were defined. Also, we assumed that the outgoing semi-trailers were equally distributed between destinations. For example, for the problem with ten incoming semi-trailers and three destinations, we had four semi-trailer departures to destination I, and three semi-trailer departures to destinations II and III, respectively. Table 2.3 presents the distribution of semi-trailers between destinations.

Table 2.3 Distribution of semi-trailers for each example problem

		Semi-trailers per destination									
# In S.T.	# dest.	I	II	III	IV	V	VI	VII	VIII	IX	X
Five	3	2	2	1							
	5	1	1	1	1	1					
Ten	3	4	3	3							
	5	2	2	2	2	2					
	10	1	1	1	1	1	1	1	1	1	1
Twenty	3	7	7	6							
	5	4	4	4	4						
	10	2	2	2	2	2	2	2	2	2	2
Forty	3	14	13	13							
	5	8	8	8	8	8	8	8	8		
	10	4	4	4	4	4	4	4	4	4	4

For each defined problem, twenty sets of sample data were randomly generated. The S.I.O.A and S.I.O.G.A were tested by generated samples. Moreover, we completely enumerated all the possible incoming and outgoing sequences for the problem with five incoming semi-trailers and ran L.U.A for each possibility to obtain the minimum and maximum profit.

Table 2.4 presents the algorithm performance for the problem with five semi-trailers. The solution for the L.U.A. is considered as an initial answer, which is used as a base to compare the average improvements for each method.

As discussed before, semi-trailer incoming and outgoing sequences, loading and unloading policies influence the direct flow of products. In this experiment, the highest and lowest profits indicate the optimal loading and unloading policy for the best and worst incoming and outgoing sequences. As presented in Table 2.4, the gap between the lowest and the highest profit is noticeable (45% and 16.2% for three and five destinations). This indicates that even with a good loading and unloading policy, the direct flow of products highly depends on the loading and unloading sequences of semi-trailers.

Second, both algorithms presented good performances (average gap between the proposed algorithms and the sequence with highest profit varied between 1.2% and 5.6%).

Considering algorithmic procedure, in S.I.O.G.A the evolutionary search is restricted to incoming sequence and after each evolutionary process ; the L.U.A is run to obtain optimal policy. On the other side, in S.I.O.A the evolutionary search is used for both sequences and for each evolutionary process it has to run L.U.A. to calculate the profit. Therefore, as presented in Table 2.4, Table 2.5 and Figure 2.7, on average S.I.O.G.A showed better performance in results and running time for all sample problems (Compare 11.1 % improvement in S.I.O.A to 13% improvement in S.I.O.G.A for the problem with three destinations in Table 2.4).

Table 2.4 Average algorithm performance for five semi-trailer sample problems

	LP(1)&HP(2)	**S.I.O.A (3)& L.U.A(4)**	**S.I.O.A & HP**	**S.I.O.G.A(5) & L.U.A**	**S.I.O.G.A & HP**
Three	43.50%	11.10%	4.40%	13.00%	2.50%
Five	16.20%	5.50%	5.60%	9.90%	1.20%

(1) L.P : Lowest profit
(2) H.P : Highest profit
(3) S.I.O.A : Sequencing incoming/outgoing algorithm
(4) L.U.A : Loading/unloading algorithm
(5) S.I.O.G.A : Sequencing incoming/outgoing greedy algorithm

Table 2.5 Average algorithm improvements (compared to initial solutions) and CPU times in milliseconds for all sample problems

		Problem size (Number of semi-trailers)							
		Five		**Ten**		**Twenty**		**Forty**	
Dest.	Alg.	Imp.	Time	Imp.	Time	Imp.	Time.	Imp.	Time
Three	S.I.O.A	11.10%	63	6.20%	148	8.30%	1618	7.40%	27284
	S.I.O.G.A	13.00%	26	7.80%	69	8.83%	618	7.70%	10703
Five	S.I.O.A	5.50%	54	11.95%	157	10.95%	4377	8.85%	119488
	S.I.O.G.A	9.90%	26	15.75%	83	12.43%	1228	10.03%	39008
Ten	S.I.O.A	N/A	N/A	6.60%	168	12.90%	2958	10.43%	306326
	S.I.O.G.A	N/A	N/A	7.55%	69	14.18%	916	12.89%	94894

2.5 Conclusion

Cross-docking is a transshipment platform in which products from incoming semi-trailers are unloaded and then consolidated with other products for reshipment. The efficiency of such platforms is highly dependent on the how the products are transferred inside them. Three factors are important for product flow : their travel distance, congestion and the moving path (a direct transfer or move to storage). In this paper, our focus is on the moving path of products. In other words, in this paper our objective is to increase the number of products that transit directly between receiving and shipping doors by ordering semi-trailers.

Two methods were proposed as a resolution approach. The first method is an integrated evolutionary algorithm with a dynamic programming function and the second one is a heuristic method. Experiments were performed on the problem with five, ten, twenty and forty incoming semi-trailers. In addition, we consider all possible incoming and outgoing sequences for the small sample problem to investigate the performance of the proposed algorithms. Overall, the heuristic approach showed better performance in time and profit (number of direct transiting products).

In this paper, we assumed that both sequences are variable, but in practice, due to platform constraints, managers have to perform some restrictions on semi-trailer sequences. In such situations, these algorithms can easily be modified to respect the circumstances.

In addition, this research focuses on the number of movements for each product. Therefore, we consider one incoming and one outgoing door. This model is not realistic. In practice there are more than one incoming and one outgoing door. This model could develop and integrate with other models such as the model proposed in (Bartholdi et Gue (2000) ;Bartholdi et Gue (2004)), to program a proper scheduling system in cross-docking platforms.

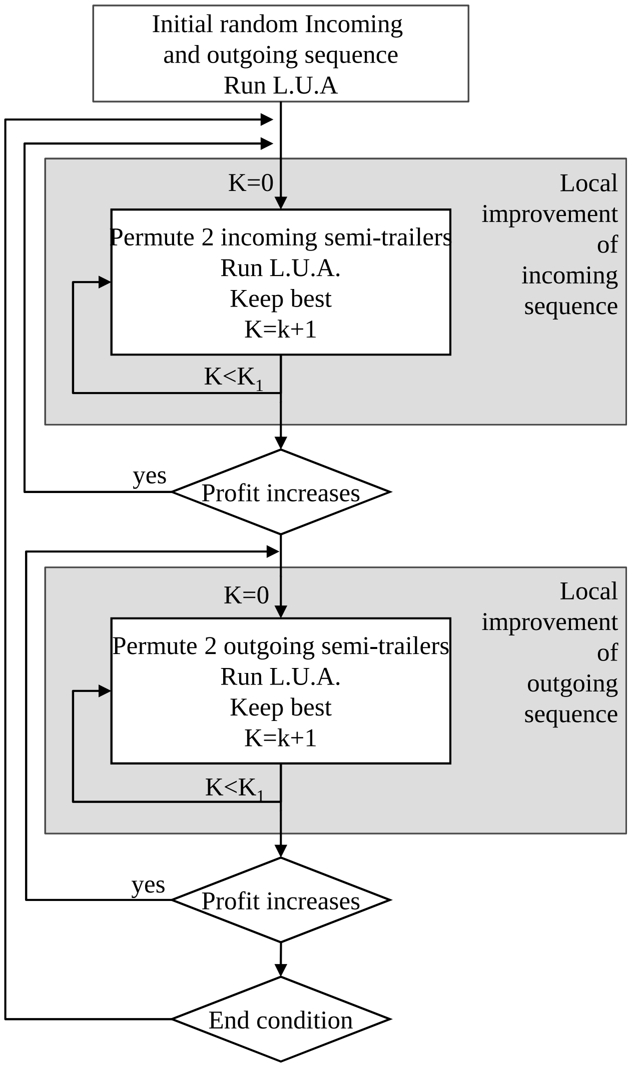
Initial random Incoming and outgoing sequence Run L.U.A
K=0
Permute 2 incoming semi-trailers
Run L.U.A.
Keep best
K=k+1
K<K1
Local improvement of incoming sequence
yes
Profit increases
K=0
Permute 2 outgoing semi-trailers
Run L.U.A.
Keep best
K=k+1
K<K1
Local improvement of outgoing sequence
yes
Profit increases
End condition

Figure 2.5 Sequencing incoming and outgoing algorithm (S.I.O.A)

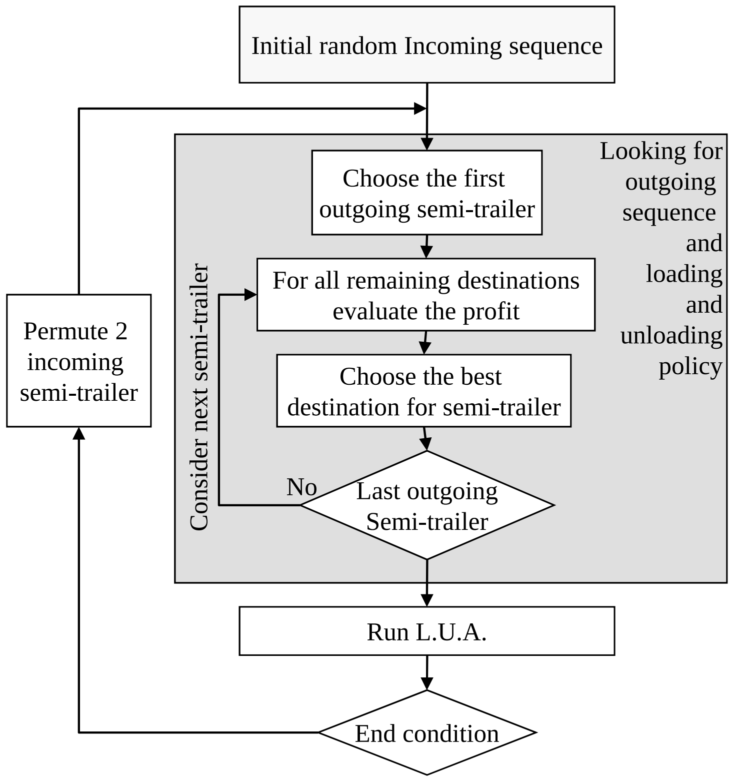
Initial random Incoming sequence
Choose the first outgoing semi-trailer
For all remaining destinations evaluate the profit
Choose the best destination for semi-trailer
Last outgoing Semi-trailer
No
Consider next semi-trailer
Looking for outgoing sequence and loading and unloading policy
Permute 2 incoming semi-trailer
Run L.U.A.
End condition

Figure 2.6 Sequencing incoming and outgoing greedy algorithm (S.I.O.G.A)

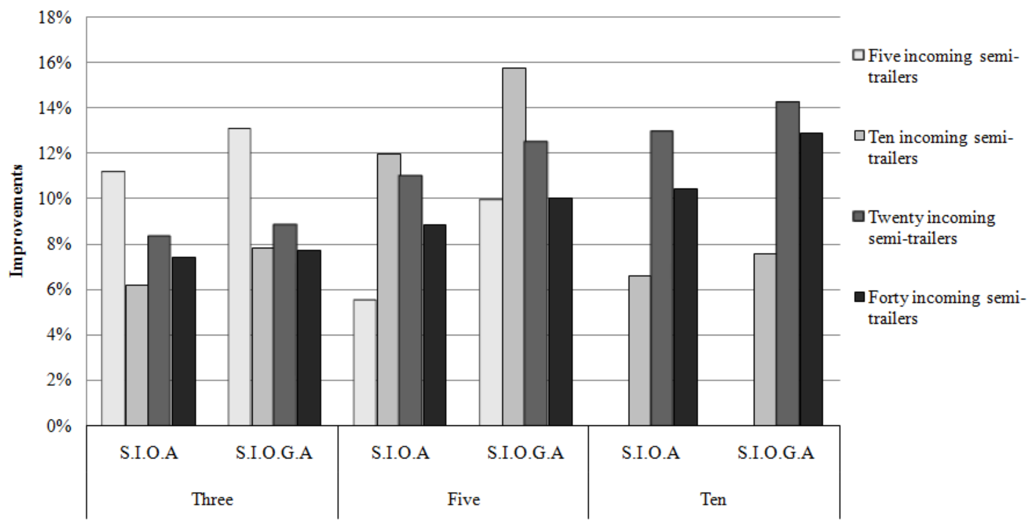

Figure 2.7 Sequencing incoming and outgoing greedy algorithm (S.I.O.G.A)

CHAPTER 3

ARTICLE 2 : AN INTEGER PROGRAMMING APPROACH TO SCHEDULING INTERNAL TRANSHIPMENT AT CROSS-DOCKS IN LESS-THAN-TRUCKLOAD INDUSTRY

Chapter Information : An article based on this chapter was submitted for publication. M.Y.Maknoon, F.Soumis, and P.Baptiste.An integer programming approach to scheduling internal transhipment at cross-docks in less-than-truckload industry.

Abstract The main obstacle of cross-docks in less-than-truckload industries is to accomplish their internal transhipments at minimum expense. This paper introduces an exact method for solving the problem of material handling at cross-docking centers. The scheduling model synchronizes the serving order of trucks with internal transfer and provides a detailed operational plan. Some families of valid inequalities are presented to strengthen the formulation. A specialized branching algorithm is developed. Several structural properties and a heuristic method are proposed to enhance the algorithm. Computational experiments of up to 40 trucks illustrate the efficiency of the developed approach.

keyword Cross-Dock , Material Handling , Scheduling , Branch and Bound

3.1 Introduction

Cross-dock is a center that transships freight between trucks with minimal use of storage in between. Cross-docking reclaims transportation efficiency by bundling arriving freight into full truckloads. Lowering the entire inventory level is another economical advantage. However, cross-docking is beneficial as long as savings in inventory and transportation expenses do not overwhelm the cost of material handling at the platform (Bartholdi et Gue (2000) ;Gue (1999)).

The problem of material handling at cross-dock deals with decisions on how to transfer freight inside the terminal. It is a process that has a direct interaction with platform key resources (i.e., operators, internal transporters). Therefore, the cost of material handling is defined based on the effective use of platform resources.In transhipment process, double handling refers to an additional retrieval and displacement for freight in temporary storage area.

It is an inefficient phenomenon that influences important platform performance indicators (e.g., handling rate, output rate, operator requirement, processing time, usage of temporary storage) (Schwind (1995)). This paper aims to present a scheduling model that reduces the cost of double handling in the transhipment process.

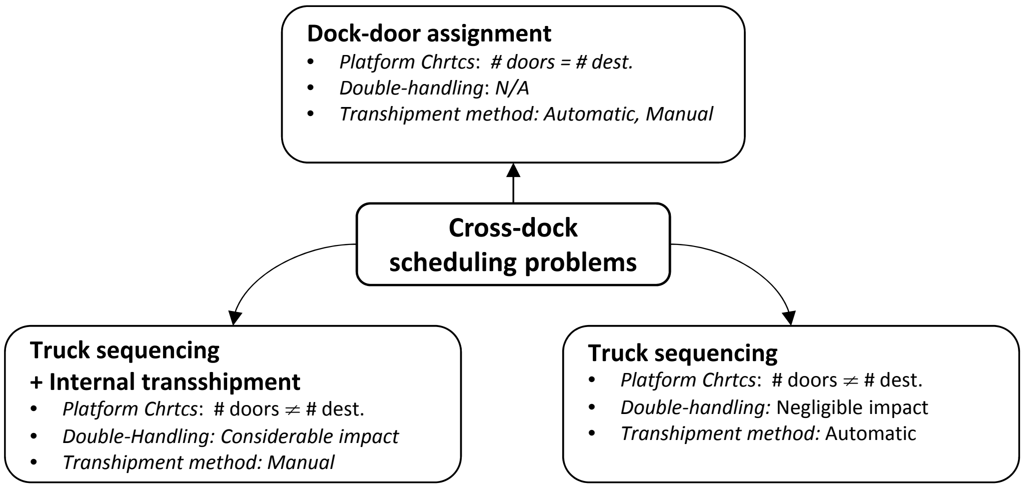
Dock-door assignment
Platform Chrtcs: # doors = # dest.
Double-handling: N/A
Transhipment method: Automatic, Manual
Cross-dock scheduling problems
Truck sequencing + Internal transsshipment
Platform Chrtcs: # doors ≠ # dest.
Double-Handling: Considerable impact
Transhipment method: Manual
Truck sequencing
Platform Chrtcs: # doors ≠ # dest.
Double-handling: Negligible impact
Transhipment method: Automatic

Figure 3.1 Cross-dock scheduling problems

According to the literature, most studies on cross-dock scheduling are quite recent (see Van Belle *et al.* (2012) ; Boysen et Fliedner (2010) for an extensive review). This research can be classified in three categories according to the impact of double-handling on the operational cost. Figure 3.1 illustrates this classification.

In the first group of studies, cross-dock has a sufficient amount of doors to process all trucks. With this platform characteristics, all of the freight directly displaced within doors.The scheduling problem focuses on optimizing total transfer and congestion. Gue (1999) and Tsui et Chang (1992) have studied platform layout design for receiving and shipping doors. For a known door layout with the same goal, much research has been performed in operational scheduling to assign trucks to platform doors (see Cohen et Keren (2009) ; Oh *et al.* (2006) ; Bartholdi et Gue (2000) ; Chmielewski *et al.* (2009) ; Lim *et al.* (2006) ; Miao *et al.* (2009)).

In the second and the third groups of studies, the platform has a limited amount of docks. Therefore, the aim of the scheduling model is to determine a processing order for the trucks.

The second group of studies focuses on truck loading and unloading schedule. The scheduling problem is to determine a sequence of vehicles at the dock to minimize total operational time (The time between unloading the first truck to loading the last one). This problem is an operational issue for distribution centers in courier industries in which highly automated conveyors and sortation systems are used to process a large volume of parcels every day. In this transhipment method, double-handling has negligible impact on total operational cost. Thus, scheduling problem is expressed as time indices models (see Yu et Egbelu (2008) ; Boysen *et al.* (2010) ; Vahdani et Zandieh (2010) ; Soltani et Sadjadi (2010) ; McWilliams *et al.* (2005) ; McWilliams (2009) ; Konur et Golias (2013)).

The third group of research develops scheduling models for cross-dock facilities in Less-than-TruckLoad (LTL) industries. In these terminals, labor intensive transporters (e.g., fork-lifts, pallet jacks) are employed. With this handling approach, product transhipment is costly and double-handling has a significant impact on the total operational expense (Bartholdi et Gue (2000)).

Unlike studies in truck sequencing for courier industries, the modeling approach based on reducing attendance time of the vehicle at the platform does not provide a precise measure of operational cost due to the following reasons :

First, the actual processing cost is directly related to the cost of each transfer operation. This means that direct transhipment or transferring via storage have different transferring costs. Even for fixed sequences of inbound and outbound trucks, various material handling decisions result in having different transhipment costs. Thus, a detailed transhipment plan is required.

Second, reducing the truck processing time at the platform dock may increase double-handling. In other words, decreasing vehicle presence at the inbound door may force the operator to transfer the remaining freight to a storage area. Also, at the outbound door, it may force the operators to use the products in storage to fill the truck. However, these products could be directly transferred if we synchronize the process of loading and unloading trucks with material handling decisions.

In literature, Boysen (2010), has investigated special LTL cross-dock in food industries with cooling requirements, that forbid internal storage. Therefore, all products are directly transferred within doors. The study considers a platform setting with single receiving and shipping dock. A dynamic programming model has been represented to minimize operational cost.

The problem of double-handling was first introduced by Maknoon et Baptiste (2009), the authors have proposed a heuristic as a resolution approach for the platform with a single receiving and shipping door. Further, for the same platform setting, the problem is proven to be NP-Hard in the strong sense. A polynomial algorithm was introduced to schedule product transhipment when the loading and unloading order of trucks is known (Sadykov (2012)). This paper is in line with previous studies, as we focus on the product displacement route. The presented scheduling simultaneously considers internal transhipment and the trucks' loading and unloading order.

In Section 3.2, we suggest a mathematical formulation that minimizes double-handling in internal transhipment. In Section 3.3, we introduce a series of valid inequalities that are added to the model. In Section 3.4, we present a path branching algorithm. Computational results are reported in Section 3.5, followed by the conclusion in Section 3.6.

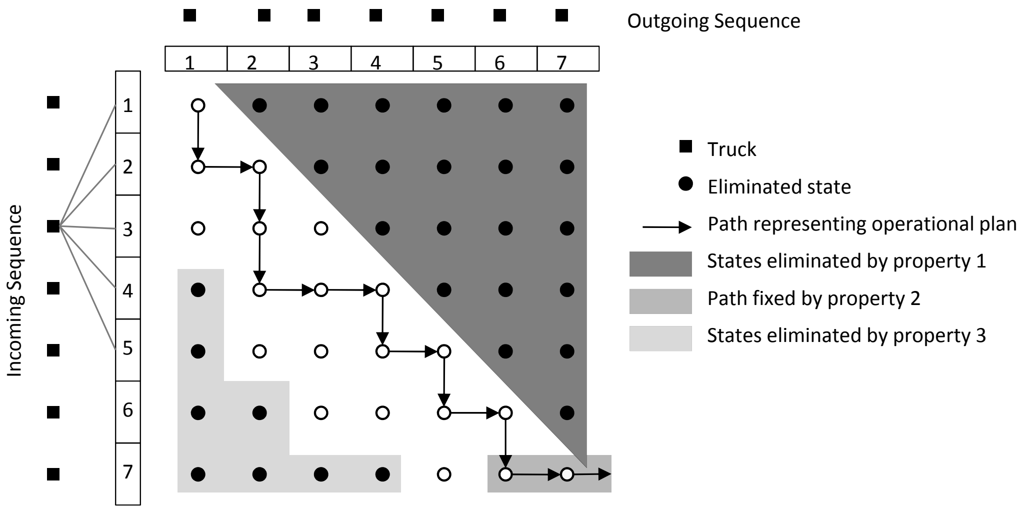
Outgoing Sequence
1
2
3
4
5
6
7
Incoming Sequence
Truck
Eliminated state
Path representing operational plan
States eliminated by property 1
Path fixed by property 2
States eliminated by property 3

Figure 3.2 Model Schema

3.2 Problem description and mathematical formulation

At the platform parking area, $2 \times T$ trucks (T incoming and T outgoing) are waiting to be assigned to the dock door. As presented in Figure 3.1, there is a processing queue of receiving ($i \in I$) and shipping ($o \in O$) trucks. Each incoming vehicle ($k \in K$) has an order for unloading (Y_i^k) and contains products for different destinations ($a^{k,d}$). Similarly, each outgoing vehicle has a processing sequence by which it is loaded for a single destination ($F^d_{(i,o)(i',o')}$). For known processing sequence of vehicles, $S_{(i,o),(i',o')}$ is a decision to either liberate the incoming or the outgoing dock. The operational plan represents a path from the first state (first incoming and outgoing order) to the last one.

For an unloaded product for destination ($d \in D$), if the outgoing truck is present at the platform door, it can be directly displaced to shipping dock. Otherwise, it is routed to a temporary storage area for future reshipment. The latter option causes double-handling.

The objective of this paper is to determine an operational plan that minimizes the total handling cost. We consider a platform with a single receiving and shipping door with no

Table 3.1 Summary of notations

$\mathbf{i} \in \mathbf{I}$	Serving order of incoming trucks	$\mathbf{o} \in \mathbf{O}$	Serving order of outgoing trucks
$\mathbf{k} \in \mathbf{K}$	Set of incoming trucks	$\mathbf{d} \in \mathbf{D}$	Set of outgoing destinations
$\mathbf{v}$	Truck capacity	$\boldsymbol{\pi}$	Path representing operational plan
$\mathbf{a^{k,d}}$	The number of products for destination d in incoming truck k		
$\mathbf{S_{(i,o),(i',o')}}$	Binary variable presenting transition from state (i, o) to state (i', o')		
$\mathbf{Y_i^k}$	Binary variable showing the assignment of truck k to order i		
$\mathbf{F^d_{(i,o)(i',o')}}$	Binary variable showing the loading process for destination d		
$\mathbf{X^{k,d}_{i,o}}$	Real variables $\in [0, 1]$, representing portion of products that are transferred directly from truck k in incoming order i, to outgoing truck in order o which departs to destination d		

restrictions on storage capacity. Moreover, a homogeneous fleet with the same capacities are considered. All trucks leave the platform either fully loaded or completely unloaded. Considering these characteristics, the model $(\mathcal{Z})$ is defined as follows :

$$\mathcal{Z} = Max \quad \sum_{i\in I}\sum_{o\in O}\sum_{k\in K}\sum_{d\in D} a^{k,d}X^{k,d}_{i,o} \tag{3.1}$$

$$\sum_{k\in K} X^{k,d}_{i,o} \le F^d_{(i,o)(i+1,o)} + F^d_{(i,o)(i,o+1)} \quad \forall i \in I, o \in O, d \in D \tag{3.2}$$

$$\sum_{o\in O} X^{k,d}_{i,o} \le Y^k_i \quad \forall i \in I, k \in K, d \in D \tag{3.3}$$

$$\sum_{i}\sum_{k\in K} a^{k,d}X^{k,d}_{i,o} \le v \sum_{i\in I} F^d_{(i,o)(i,o+1)} \quad \forall o \in O, d \in D \tag{3.4}$$

$$\sum_{i'=1..i} a^{k,d} \times Y^k_{i'} - v \sum_{i'=1..i}\sum_{o\in O} F^d_{(i',o)(i',o+1)} \ge 0 \quad \forall i \in I, d \in D \tag{3.5}$$

$$\sum_{k\in K} Y_{k,i} = 1 \quad \forall i \in I \tag{3.6}$$

$$\sum_{i\in I} Y_{k,i} = 1 \quad \forall k \in K \tag{3.7}$$

$$S_{(i,o)(i',o')} = \sum_{d\in D} F^d_{(i,o)(i',o')} \quad \forall (i \in I, o \in O), (i' \in I, o' \in O) \tag{3.8}$$

$$S_{(i-1,o),(i,o)} + S_{(i,o-1),(i,o)} = S_{(i,o),(i+1,o)} + S_{(i,o),(i,o+1)} \quad \forall (i \in I, o \in O) \tag{3.9}$$

$$S_{(0,0),(1,0)} = 1 \tag{3.10}$$

The objective (3.1) maximizes direct transhipment within doors (reduces total handling cost). Constraints (3.2) and (3.3) express the displacement decision. The limits on the number of direct transferred products for each truck is satisfied by (3.4). Constraint (3.5) imposes full load assumption on outgoing trucks. Constraints (3.6) and (3.7) are assignment decisions for incoming trucks. Finally, constraints (3.8)-(3.10) link the transition states and ensure the existence of the path.

3.3 Valid inequalities

The model ($\mathcal{Z}$) can be strengthened by the set of valid inequalities. These inequalities restrict the direct transhipments between receiving and shipping vehicles.

The first set of inequalities is a loose form of flow conservation constraint for variable $F^d_{(i,o)(i',o')}$. It makes sure that an outgoing truck (o) only loads products for one destination.

Proposition 1. *For any pair of states ($i \in I$, $o \in O$) and destinations ($d \in D$) the following inequality holds :*

$$F^d_{(i-1,o)(i,o)} \leq F^d_{(i,o)(i+1,o)} + F^d_{(i,o)(i,o+1)} \quad \forall (i \in I, o \in O), d \in D \tag{3.11}$$

For a given sequence of trucks, there are several symmetrical transferring decisions that have the same outcomes in terms of direct transhipment. Proposition 2 represents a class of valid inequalities to limit symmetrical possibilities.

Proposition 2. *The following inequalities are valid for model (3.1-3.10)*

$$\sum_{o \in O} F^d_{(i,o)(i+1,o)} \leq \sum_{o} \sum_{k} X^{k,d}_{i,o} \quad \forall i \in I, d \in D \tag{3.12}$$

Démonstration. When the shipping truck leaves the platform, it carries all the products inside the current unloading vehicle assigned to destination d. If it is not possible to transfer all the freight, it means that the shipping truck is already loaded with all the products inside previous loading ones. That is, there are many combinations of direct transhipment decisions that are equivalent (all of the combinations surpass the truck capacity). As a result, we break many symmetrical solutions by assuming that all products from the current incoming vehicle are directly displaced. □

Proposition 3. *For an incoming sequence order ($i \in I$) and destination ($d \in D$) the following* *inequality holds :*

$$\sum_{i'=1..i}\sum_{k\in list} Y_{i'}^{k} + \sum_{i'=1..i}\sum_{o\in O} F_{(i',o)(i',o+1)}^{d} \leq i + B_{i,m}^{d} \quad \forall i \in I, m \in list, d \in D \tag{3.13}$$

This inequality imposes an upper bound on the number of shipping vehicles for destination ($d \in D$) from the beginning to the current unloading sequence ($i \in I$) (shown by $B_{i,m}^{d}$). Algorithm 1 presents the procedure of generating the inequalities which is described by the following example.

Algorithm 1 Generating inequality (3.13)

```
Input : Model $\mathcal{Z}$
Output : Valid inequality (13)
1 :  for all destinations ($d \in D$) do
2 :    Sort all arriving trucks ($k \in K$) in ascending order
       based on their content for destination ($d \in D$)
3 :    for all incoming sequences ($i \in I$) do
4 :      for all positions in sorted list ($m$) do
5 :        $B_{i,m}^{d}$ ←calculate maximum allowable outgoing shipments
             when $i$ trucks are unloaded and first arriving truck
             has position $m$ in the sorted list
6 :        if $B_{i,m}^{d} \neq B_{i,m-1}^{d}$ then
7 :          Build a list of trucks
8 :          Write inequality (13)
```

Example : Suppose that 6 incoming vehicles with a capacity of 10 products each, are present at the platform. As shown in Table 3.2, each truck contains products for three destinations (Named A,B,C). For example, truck (i) has 6 products destined to A, 3 to B and 1 to C.

For shipping destination "A", first, we sort vehicles based on content assigned to "A" in descending order. Table 3.3 reports the sorted order of trucks for all destinations. Consider a case in which two ($i = 2$) vehicles are unloaded. In this situation, at most one truck could depart for "A" (that means $B_{2,1}^{A} = 1$). Therefore, we write the constraints as follows :

$$\sum_{i=1..2}\sum_{k\in List} Y_{k,i} + \sum_{i=1..2}\sum_{o\in O} F_{(i,o)(i,o+1)}^{A} \leq 3 \qquad \textbf{\textit{List}}=\{ii,i,v,iv,vi,iii\}$$

However, if trucks (ii), (i) are not unloaded in the first and the second sequences, then, the maximum number of direct transhipment obtained by unloading trucks (v) and (iv) is equal to 9. In this situation, we are unable to depart any vehicle for "A" ($B_{2,3}^{A} = 0$). The same fact is also valid for (vi) and (iii). Therefore, the following constraint is added :

$$\sum_{i=1..2}\sum_{k\in list} Y_i^k + \sum_{i=1..2}\sum_{o\in O} F^A_{(i,o)(i,o+1)} \leq 2 \qquad \boldsymbol{List}=\{v,iv,vi,iii\}$$

A similar procedure is repeated for all other incoming trucks and outgoing destinations, which is described in Algorithm 1

3.4 Path-based branching algorithm

We devise a path-based branching algorithm on model ($\mathcal{Z}$) to solve the problem of material handling to optimality. First, valid inequalities are added to the model. Second, pre-processing is performed to remove some states from the model. Finally, the algorithm starts with an initial solution which is found by our heuristic model. Algorithm 2 presents the schema of this approach.

Algorithm 2 Branching schema

```
INITIALIZE
I1 :   Add valid inequalities
I2 :   Perform pre-processing
I3 :   $L_F \leftarrow$ Apply feasibility check
I4 :   $\underline{Z}_{Best} \leftarrow$ Run path-diving heuristic
PATH-BRANCHING
B1 :   repeat
B2 :     $\hat{o} \leftarrow$ Select an outgoing order from remaining candidate
             (Outgoing which has minimum cumulative branching pseudo cost)
B3 :     repeat
B4 :       $\hat{i} \leftarrow$ Select an incoming order associated with selected outgoing $\hat{o}$
             (Incoming with minimum branching pseudo cost is selected)
B5 :       $\Pi \leftarrow S_{(\hat{i},\hat{o})(\hat{i},\hat{o}+1)}$
B6 :       if selected variable is in $L_F$ then Solve $F_\Pi$
B7 :       if $F_\Pi$ is not feasible or relaxation is less than the best solution then
B8 :           Exclude variable from the path $\Pi$ Goto B3
B9 :       if there exists a path then Go to (B10) else Go to (B1)
B10 :      Branch on sequencing variables (i.e. $Y_i^k$ and $F_{(i,o)(i',o')}$)
B11 :    until no candidate is remained
B12 :  until no candidate is remained
```

3.4.1 Pre-processing

In this section, we introduce properties that are applied in model ($\mathcal{Z}$) to revoke some defined states that are not valid in the optimal solution. These properties are based on permitted connections between incoming and outgoing sequences.

Property 1. *All states in which the number of shipping trucks is more than the number of receiving ones are not feasible.*

Table 3.2 Sample instances to demonstrate the algorithm used to write constraint (3.13)

	Truck					
Destination	**i**	**ii**	**iii**	**iv**	**v**	**vi**
A	6	8	3	4	5	4
B	3	2	4	2	3	6
C	1	0	3	4	2	0

Table 3.3 Sorted order of trucks based on the number of products

	Sorted order					
Destination	1	2	3	4	5	6
A	ii	i	v	iv	vi	iii
B	vi	iii	v	i	iv	ii
C	iv	iii	v	i	vi	ii

Démonstration. Each shipping truck departs the platform fully loaded. As all vehicles have the same capacity, at least T inbound vehicles have to be unloaded to send T shipping truck. Therefore, all states that violate this condition are eliminated. □

Property 2. *All freight in the final unloading trucks are directly transferred.*

Démonstration. Based on the problem assumptions, all products are sent to their destinations at the end of the planning horizon. Accordingly, all content of the last incoming truck is directly loaded to the last departing vehicle for each destination. Therefore, all associated transition states are selected. □

Property 3. *In the optimal operational path, each incoming truck can be used to load at most $(D+1)$ outgoing shipping vehicles. Similarly, for an outgoing truck the amount of unloading vehicles is bounded.*

Démonstration. All products in the unloading vehicle can be directly loaded to D empty outgoing trucks. However, if the current shipping truck, to destination d, is partially loaded, then products for that destination are shipped by at most 2 trucks. If we load more than $(D+1)$ shipping vehicle, the incoming truck is empty and replacing it does not violate the optimal solution.

Similarly, the direct transhipment is bounded by the capacity of an outbound truck. As soon as the shipping vehicle is fully loaded with products that are directly transferred, it can be replaced by another one. Thus, we could trace a bound on the number of unloading transitions by analyzing the worst arrival assignment. □

Remark : By using properties (1-3), we are able to exclude some states from the branching. In addition, we eliminate additional states based on the solution of the heuristic approach. We start from the first outgoing order and solve the relaxation problem for each corresponding incoming order. The state is removed from the branching region if the relaxed solution is not better than the heuristic one.

Example : Figure 3.1 represents the eliminated states after applying properties 1, 2 and 3. In this representation, we assume that trucks ship to two destinations that are fully loaded when at most three incoming vehicles are unloaded. By using the second property, we can fix the value of transition states $S_{(7,6)(7,7)}$ and $S_{(7,7)(7,8)}$ to one. Whereas the third property eliminates states $(7,1)-(7,4)$ and $(6,1)$ as they need more than 3 transitions. Similarly, states $(4,1)$ to $(7,1)$, $(6,2)$, $(7,2)$ are eliminated due to the bound on the loading trucks.

3.4.2 Testing the feasibility of the operational plan

The relaxation problem provides a bound on maximum direct transhipment. However, it does not have any information about the feasibility of the operational plan. In branching, this is critical as some infeasible paths exist (for the selected path, there are no assignments that satisfy a full truck load condition). To avoid these situations, we utilize constraints (3.5-3.10) of model ($\mathcal{Z}$) , we call it model ($\mathcal{F}$), to evaluate the feasibility of the branched path (Π). This model ($\mathcal{F}$) has fewer variables and is relatively easy to solve ; however, using it at each branching step will cause computational burden.

Definition : A path Π is inferior to Π' if and only if, for all transition variables ($S_{(\hat{\imath},\hat{o})(\hat{\imath},\hat{o}+1)}$) in Π, there exists a relative transition state in Π' for outgoing order whose incoming order is greater than or equal to $\hat{\imath}$.

Observation 1. *Π^* is a path representing an operational plan in which we alternate incoming and outgoing orders. Clearly, all feasible paths are inferior to Π^*.*

Proposition 4. *If Π' is feasible, then all Π inferior to Π' are also feasible.*

Démonstration. When Π' is feasible, it means that there exists a sequence of incoming and outgoing trucks which is feasible in ($\mathcal{F}$). These orders are also valid for all Π inferior to Π' as we only unload more products from the incoming trucks. $\square$

Observation 2. *If Π^* is feasible then all states are also feasible.*

Based on the above properties, before starting the branching, we try to determine a path near Π^* that is feasible. We start from the first state and follow the direction of Π^*. We employ model ($\mathcal{F}$) to evaluate the feasibility of the partially constructed path. If it is infeasible, the algorithm backwards one state and adds another alternative. The process terminates at the final state. The providing approach constructs a feasible path near Π^*. All the states above this path are added to the list (L_F). This list represents the states that may not have a feasible solution. During the branching process, we call model ($\mathcal{F}$) to verify their feasibility.

3.4.3 Path-diving heuristic for initial solution

In the branching algorithm, the path-diving heuristic is used to obtain an initial solution. This method is based on the information that is obtained by solving the linear relaxation.

Given the optimal solution of the relaxation problem, the heuristic follow a diving strategy to construct a feasible path. It starts from the first state, and iteratively examines the objective value for each candidate and selects the best one. During this process model ($\mathcal{F}$) is employed to verify the feasibility of the path. Then, for given path it looks for the best truck assignment.
Finally, a refined procedure is employed by backtracking and verifying the states that have the same relaxation solution.

3.4.4 Branching strategy

The branching is decomposed into two phases. First, we only branch on variables that are a part of the path. We call upon model ($\mathcal{F}$) to verify the feasibility of the path. Clearly, a variable is discarded if the path is infeasible. Second, we branch on sequencing variables with the priority on the outgoing sequences.

In the first phase, the structure of the path enables us to limit the branching candidate to the arcs of the form $S_{(i,o)(i,o+1)}$, also we only fix the variables to their upper bound (branch up). Instead of branching variables to their lower bound, we branch up on other alternatives.

In model ($\mathcal{Z}$) , each state (i,o) has a value that represents the number of direct transshipments ($A^{k,d}X^{k,d}_{i,o}$). Due to the path structure, selecting transition variables excludes some states from the branching. For example, if we select the variable ($S_{(\hat{i},\hat{o})(\hat{i},\hat{o}+1)} = 1$) then, all transition variables that have the condition of ($i > \hat{i}, o < \hat{o}$) or ($i < \hat{i}, o > \hat{o}$) are excluded from branching.Therefore, we calculate a pseudo cost value which represents the marginal loss in the value of objective function if the variable ($S_{(\hat{i},\hat{o})(\hat{i},\hat{o}+1)} = 1$) is selected.

$\boldsymbol{List}_{(\hat{i},\hat{o})}$:$\{(i,o)|(i > \hat{i}, o < \hat{o})\ or\ (i < \hat{i}, o > \hat{o})\}$

$$C_{(\hat{i},\hat{o})} = \frac{\sum\limits_{(\hat{i},\hat{o})}\sum\limits_{k}\sum\limits_{d} A^{k,d}X^{k,d}_{i,o}}{1-\bar{S}_{(i,o)(i,o+1)}} \quad \forall k \in K, d \in D, (i,o) \in List_{(\hat{i},\hat{o})} \tag{3.14}$$

Each $C_{(\hat{i},\hat{o})}$ represents the possible loss in objective function while increasing the value of $\bar{S}_{(i,o)(i,o+1)}$ to one.

The candidate is selected in two steps : first, we choose the outgoing order, i.e., the order with the lowest cumulative pseudo cost is chosen. Then, for the selected outgoing order, a variable with minimum pseudo cost is selected for branching.

3.5 Computational experiments

The computational results have been carried out on a computer with 2.2 GHz CPU and 8 GB of RAM and 4 cores. The algorithm is run with a single thread. The time limit has been set to 24 hours. We determined this time limit to demonstrate the effectiveness of initializing and branching steps in computation time. We have considered a 20 minutes computational time for the heuristic algorithm and 60 seconds for feasibility model. The path-branching algorithm is implemented in C++ and we use CPLEX 12.2 as a solver.

3.5.1 The test instances

The algorithm has been executed on the samples of three generated data sets. Each set is distinguished by the number of trucks at the platform. Samples are presented as $(T|D|C|T^d)$ where T is the total number of trucks (24, 32, 40), D is the number of destinations (2, 3, 4), C is the truck capacity (10,20) and T^d is the number of outbound trucks per destination. For each sample, we have generated 5 instances. Overall 225 instances were tested. Table 3.4 represents the characteristics of the data sets.

Table 3.4 characteristics of the data set

	T	**C**	**D**	**Truck by destination (T^d)**		
				1	**2**	**3**
			2	6,6	8,4	9,3
Set 1	24	10-20	3	2,4,6	3,6,3	4,4,4
			4	2,3,3,4	2,4,4,2	3,3,3,3
			2	6,10	7,9	8,8
Set 2	32	20	3	5,5,6	6,6,4	7,6,3
			4	4,4,4,4	5,3,4,4	6,4,3,3
			2	10,10	7,13	9,11
Set 3	40	10-20	3	5,7,8	6,6,8	7,7,6
			4	4,4,6,6	5,5,5,5	8,4,4,4

T : Number of incoming trucks

C : Truck capacity

D : Number of destinations

3.5.2 Impact of improvement strategies on solving procedure

Table 3.5 summarizes the marginal contribution of steps I1 to I4 of the initializing and the steps B1-B12 of the path-branching algorithm. Relaxation gap, number of branched nodes, and run time are chosen as performance indicators. As no special solution method was available for this problem, we chose CPLEX as a benchmark to assess the quality of the path-branching algorithm. The second set of instances with 32 trucks was selected for comparison.

The table is split in sections representing steps (column "Steps") and divided further for each sample. Rows "Node" and "CPU Time" represent the average and maximum observed values for the number of branched nodes and run time. Row "Gap" shows an average percentage gap between LP relaxation (LPSOL) and the best integer solution (IPSOL). The gap is computed as $(LPSOL - IPSOL)/100$.

The first set of rows (labelled "$\mathcal{Z}$"") provide computational results for model ($\mathcal{Z}$) by using CPLEX. as noted in the table, CPLEX could not solve 28 instances out of 40 in one day. It succeeds in solving all tests with 2 destinations ; however, the running time dramatically varies between instances. The average integrality gap is 10.25 %.

The second set of rows (marked "$\mathcal{Z}$+VI") exhibits results after adding valid inequalities on model ($\mathcal{Z}$). This time the average computational time decreases from 16 to 8 hours. The relaxation gap is reduced around 1%. However, 5 instances are remain unsolved. This indicates valid inequalities cause significant reduction in computational time, but they are less effective in decreasing the relaxation gap.

The third and forth rows (labelled "$\mathcal{Z}$+VI+Pre" and "$\mathcal{Z}$+VI+Pre+H") summarize marginal contributions of pre-processing and initial heuristics (steps I1-I4 of algorithm 3.4).

Properties 1, 2 and 3 succeed with a 5% reduction in the relaxation gap. Among the instances, (32|4|20|4, 4, 4, 4) has noticeable improvement (from 10.8% to 0.8%). An additional 1 % decrement in the relaxation gap is reported after applying path-diving heuristic and additional variable fixing. Regarding the computational time, CPLEX succeeds in solving all the instances in less than 2 hours on average. The number of branched nodes and running time is significantly decreased.

Finally, the last set of rows "Path Br.Alg." depict results of using the proposed branching approach. With the suggested strategy, CPLEX solves all the instances in less than 6 minutes on average. The number of branching node is dramatically reduced (from 9941.72 to 1311.96 on average) which demonstrates the effectiveness of the proposed branching approach.

To summarize, the path-branching algorithm enhances the defined performance indicators in the following manner : the relaxation gap reduces from 10.25 % to 3.62 %, computational time decreases from 57104.3 to 366.42 seconds and finally, the amount of branched nodes drops from 22441.18 to 1311.96.

Table 3.5 Numerical results

Steps			Instances									
			(32|2|20|6,10)	(32|2|20|7,9)	(32|2|20|8,8)	(32|3|20|5,5,6)	(32|3|20|6,6,4)	(32|3|20|7,6,3)	(32|4|20|4,4,4,4)	(32|4|20|5,3,4,4)	(32|4|20|6,4,3,3)	Average
$\mathcal{Z}$	**Gap**		20.1%	10.4%	3.8%	6.8%	8.0%	10.2%	10.8%	10.2%	12.0%	10.25%
	Node	**Max**	122581	11068	9006	51236	42032	33546	26363	25590	25371	
		Ave	49554.8	7376.8	4377.2	34818	28080.4	26443	18984.6	14394.6	17941.2	22441.18
	CPU Time[5]	**Max**	51253.8	5363.38	3464.86	>86400[1]	>86400[2]	>86400[2]	>86400[2]	>86400[1]	>86400[2]	
		Ave	21722.77	3826.03	1933.12	70359.51	>86400	>86400	>86400	70453.57	>86400	57104.3
$\mathcal{Z}$+VI	**Gap**		17.7%	8.4%	1.8%	6.8%	8.0%	10.2%	10.8%	10.2%	12.0%	9.55%
	Node	**Max**	7728	3565	122994	25525	16010	20617	12413	25299	22111	
		Ave	5068.4	2689.2	25780.6	8161.6	9077.6	13215.6	3385	7625.6	11244.4	9583.12
	CPU Time[5]	**Max**	9702.28	6411.79	72204	>86400[3]	40887.5	41423.6	>86400[4]	70655	>86400[4]	
		Ave	7048.12	5305.99	16682.54	32234.16	27989.34	28489.12	23628.55	31195.97	67644.06	26690.87
$\mathcal{Z}$+VI+Pre	**Gap**		15.3%	7.3%	1.8%	0.8%	3.4%	5.8%	0.8%	0.6%	4.0%	4.41%
	Node	**Max**	1731	1105	115143	34733	18475	16880	323660	54070	104501	
		Ave	1367.6	744.6	25521.6	13117.8	14382.2	11713.6	70762.4	17109.6	71563.6	25142.56
	CPU Time[5]	**Max**	147.95	68.11	3223.65	5185.5	2876.59	2744.88	75544.6	12671.7	25386.2	
		Ave	116.79	62.26	792.99	1979.53	2293.69	1885.68	17332.52	3990.49	17192.76	5071.86
$\mathcal{Z}$+VI+Pre + H	**Gap**		15.0%	5.6%	1.2%	0.8%	2.6%	5.0%	0.2%	0.6%	1.6%	3.62%
	Node	**Max**	1401	527	29130	9682	33177	31685	5730	121484	49871	
		Ave	1072.4	510	6484.4	2324.2	14328.2	17049	1343.2	28609.4	17754.6	9941.72
	CPU Time[5]	**Max**	128.47	35.91	2503.25	973.71	3904.38	3030.66	2286.81	30289.1	9044.65	
		Ave	114.55	31.92	595.2	281.7	1756.17	1841.49	552.13	7781.48	3696.02	1850.07
Path Br. Alg.	**Node**	**Max**	310	20	28185	757	1149	1513	317	1824	6052	
		Ave	236.8	17.6	6249	221	567.6	1087	63.8	955.4	2409.4	1311.96
	CPU Time[5]	**Max**	34.37	3.72	2321.71	185.23	342.95	494.99	212.93	1016.44	3837.67	
		Ave	25.24	2.29	539.26	52.41	160.75	335.84	43.18	606.09	1532.75	366.42

[1] Four instances were not solved

[2] All the instances were not solved

[3] One of the instances was not solved

[4] Two instances were not solved

[5] CPU time in seconds

3.5.3 Result of test instances

Table 4.6 represents a detailed analysis of the path-branching algorithm on the first and the third data set. Column headings are : instance name, number of branched nodes, branching time (CPU time in second), integrality gap, the number of inequalities generated by algorithm 1 (column "Alg.1"), and, the number of times that the algorithm could not discard the path and had to branch on sequencing variables Y_i^k and $F_{(i,o)(i',o')}$ (column "$\mathbb{SV}$"). The last two columns report run time and the gap between the initial (solution of path-diving heuristic) and the optimal solution.

Path-branching algorithm succeeds in solving 172 out of 180 instances to optimality. Instances of the first data set are solved in 13 seconds on average. The computational time considerably increases to 11509.9 seconds for the third data set (Instances with 40 trucks). Also, the instances become harder to solve as the number of destinations increases to four.

The average integrality gap for the first and the third data set are about 3.2% and 4.2% respectively. Comparing the gap between the initial and optimal solution, one can remark that most of the time the heuristic solution is also the optimal one (the average difference is 0.1% for the first set and 0.4% for the third set). Therefore, by having a time limit of 20 minutes, it is possible to determine a very good operational plan.

Column "Alg.1" represents the average number of inequalities that are generated by algorithm 1. The number of added inequalities (13) reasonably increases with the size of the model. For example, in the first data set on average 56.6 constraints are added to the model, while 125.2 inequalities are added in the third data set.

Finally, column "$\mathbb{SV}$" reports the amount of times that the algorithm starts branching on sequencing variables. For samples with 2 destinations, the algorithm mostly discards nodes before branching on sequencing variables. In samples with 3 destinations, this value is notably increases. For instances with 4 destinations, this number dramatically varies, ranging from 0 to 3615.4 on average (the third data set).

To summarize, the computational results demonstrate that the presented approach is able to solve the samples of up to 40 trucks in a reasonable time. Valid inequalities make a significant reduction in solution time, but they have less impact on the integrality gap. The properties applied in the pre-processing bring additional efficiencies to the solution process. The path-branching strategy is an effective method to optimally solve the problem by avoiding symmetries that occurred by branching on sequencing variables.

Table 3.6 Results for instances with 24 and 40 trucks

	Algorithm 1							**Heuristic**	
Instance	**Node**		**Branching Time**		**Gap**	**Alg.1**	$\mathbb{SV}$	**Gap**	**Time**
	μ	**Max**	μ	**Max**					
(24|2|10|6,6)	464.4	1739	10.7	33.2	1.2%	44	0.8	1.2%	15.4
(24|2|20|6,6)	686	1500	17.6	29.7	1.4%	34.8	1	1.4%7	20.2
(24|2|10|8,4)	13.6	16	0.8	1	2.6%	41.6	1	2.6%	3.7
(24|2|20|8,4)	54.2	99	2.3	4	10.2%	36.4	1	10.2%	5.2
(24|2|10|9,3)	69.8	210	2.4	6.6	4.2%	39	1	4.2%	5.5
(24|2|20|9,3)	105.2	143	4	5.1	12.6%	34.8	1	12.6%	7.7
(24|3|10|2,4,6)	62	100	7.4	11.1	2.6%	64.6	2.2	2.6%	51.2
(24|3|20|2,4,6)	217	691	13.1	31.7	6.6%	52.8	3.8	6.6%	40.3
(24|2|10|3,6,3)	125.4	415	14.7	50.9	3.0%	61.8	6.8	3.0%	73.8
(24|2|20|3,6,3)	128.2	188	13	20.2	5.6%	52.4	3	5.6%	42.9
(24|2|10|4,4,4)	8	23	0.7	2.3	0.4%	62.2	0.4	0.4%	32.8
(24|2|20|4,4,4)	34.4	106	4.1	12.7	0.6%	55	3.4	0.8%	15.6
(24|2|10|2,3,3,4)	30.2	89	6	17.4	0.6%	80.2	0.2	0.8%	196.5
(24|2|20|2,3,3,4)	183.2	244	31.3	43.9	1.6%	66.2	6	2.0%	20.4
(24|2|10|2,4,4,2)	16.8	52	3.7	11.9	0.4%	74.8	0	0.4%	196.1
(24|2|20|2,4,4,2)	174.2	330	30.9	55.4	1.8%	67.6	7.2	2.2%	72.6
(24|2|10|3,3,3,3)	174	812	39.4	183.3	1.8%	83	28.4	2.0%	345
(24|2|20|3,3,3,3)	279.8	1349	37.7	177.5	0.0%	68.2	5.2	0.4%	13.6
Average	**157**		**13.3**		**3.2%**	**56.6**	**4**	**3.3%**	64.3
(40|2|10|7,13)	1496.6	4609	1043.4	2842	0.8%	93.6	0.6	0.8%	669.3
(40|2|20|7,13)	3283.8	6247	3237.6	4502.3	1.8%	85.6	1	2.4%	923.6
(40|2|10|9,11)	6005.4	8652	1340.8	2134.9	9.6%	94.6	13.6	9.6%	678.1
(40|2|20|9,11)	11014.8	15228	2427.4	3488.1	22%	78.4	10.6	22%	442.6
(40|2|10|10,10)	1015.4	2160	237.6	499.8	8%	102	1	8.0%	295.1
(40|2|20|10,10)	2465	3237	603.2	754.5	21.4%	82.8	2	21.4%	264.7
(40|3|10|5,7,8)	531.6	1818	516.1	1739.7	0.2%	140.8	29.6	0.6%	996.1
(40|3|20|5,7,8)	32208.2	58045	23987.1	43477.5	3%	121.4	1630	3.4%	995.1
(40|3|10|6,6,8)	210.4	919	153.8	684.1	0.6%	139.8	10.8	0.8%	659.3
(40|3|20|6,6,8)	5076.2	14470	3430.4	8750.5	1.4%	117	210	1.8%	695.5
(40|3|10|7,7,6)	664.2	2959	558.9	2473.5	0.6%	145	23.2	1.2%	568.3
(40|3|20|7,7,6)	2304.2	5230	1648.8	4068.1	1.4%	122.8	119	1.6%	416
(40|4|10|4,4,6,6)[1]	27104	67907	34579	86400	0.8%	183.8	2908	1%	1200
(40|4|20|4,4,6,6)[1]	28240.2	78133	34579	86400	0.6%	162.4	0.4	0.8%	1067.8
(40|4|10|5,5,5,5)	4810.2	14234	9623.2	28666.4	0.8%	179.8	27.6	1.2%	1200
(40|4|20|5,5,5,5)	0	0	0	0	0%	148.2	0	0%	428.5
(40|4|10|8,4,4,4)	7879.6	25856	16323.1	45717.4	0.2%	107.2	465.7	1.6%	1200
(40|4|20|8,4,4,4)[2]	57065.6	76877	72888.2	86400	3%	148.6	3615.4	3.8%	1200
Average	**10632**		**11509.9**		**4.2%**	**125.2**	**503.8**	**4.6%**	**772.3**

[1] Two instances were not solved

[2] Four instances were not solved

3.6 Conclusion

Double-handling is an important and costly part of internal transhipment for cross-docks in LTL industries. In this paper, we provide a mathematical model to schedule platform internal transhipment. This modelling approach has two advantages : first, it directly reduces the material handling cost, and second, the outcome of scheduling minimizes total processing time (time to transfer freight). In addition, this model can be easily adopted to real life problems.

We introduce some families of valid inequalities to strengthen the relaxed model. Special path-branching schema are proposed to solve the problem. Several properties are proposed to improve the relaxation gap. Finally, this algorithm is equipped with a path-diving-heuristic to solve to optimality instances of up to 40 trucks.

In this problem, we have solved more complicated cases in which all of the trucks are available at the beginning of the planning horizon. However, in reality the arrival and departure of trucks is determined by the scheduling on a logistic network. The proposed model has the ability to cope with this situation. In addition, by restricting the arrival and departure order of trucks, we are even able to solve larger instances within a reasonable time.

CHAPTER 4

ARTICLE 3 : A SEQUENTIAL PRIORITY-BASED HEURISTIC FOR SCHEDULING MATERIAL HANDLING IN A SATELLITE CROSS-DOCK

Chapter Information : An article based on this chapter was submitted for publication. M.Y.Maknoon, O.Kone, and P.Baptiste, A Sequential priority-based heurisitc for scheduling material handling in a satellite cross-dock.

Abstract In a less-than-truckload network, the satellite cross-dock is in charge of local deliveries. These terminals operate in two separate shifts : consolidating pickup freight for overnight shipments and processing received products for early morning delivery. Therefore, the scheduling priority is to accomplish internal transhipment with minimum handling costs. In this paper, we formalize the handling process and present a mathematical model to schedule internal transfer with minimum handling cost. We also present a sequential priority-based heuristic to tackle the practical problem. Numerical results depict the stability of the heuristic method for a fairly large size.

keyword Less-than-truckload , Satellite cross-dock , Material handling , Mixed-integer programming , Heuristics

4.1 Introduction

A cross-dock is a distribution center with the function of consolidating arriving freight with the same destination in order to have full outgoing truckloads. Cross-docking is widely practiced in Less-than-Truckload (LTL) shipping to improve the economy of scale in transportation (Apte et Viswanathan (2000)). Furthermore, it reduces the total inventory level and finds savings in storage costs. However, cross-docking is beneficial as long as the handling costs do not overwhelm the savings in transportation and inventory costs (Bartholdi et Gue (2004)) .

Generally, the LTL network employs a hub-and-spoke arrangement to shift freight. In this strategy, a satellite terminal is responsible for local deliveries. The special working structure of these terminals provides flexibility with the network timing schedule. They operate in two separate shifts to process products. Arriving freight is processed for early morning deliveries, whereas products that have been picked up are consolidated for overnight shipments. Outside of these two periods of time, the terminal is inactive (Bartholdi et Gue (2000) ; Gue

(1999)). Therefore, the main objective is to boost operational efficiencies by examining total operational cost, since the platform has flexibility with its truck scheduling.

Research on operational processes of cross-docks can be classified in two categories according to the internal transhipment system : automated and manual systems. Some platforms are equipped with highly automated conveyors and sortation systems (e.g., processing centers in courier industries). Studies for this type of center deploy a time related objective (e.g., total operational time, tardiness of outbound truck, etc.,) to synchronize the loading and unloading process. For a cross-dock with a single receiving and shipping door, Yu et Egbelu (2008) and Boysen *et al.* (2010) have represented a heuristic method to schedule the arrival and departure of trucks in order to minimize total operational time. This problem has been studied in (Vahdani et Zandieh (2010) ; Soltani et Sadjadi (2010) ; Boloori Arabani *et al.* (2011) ; Larbi *et al.* (2011)), in which several meta-heuristics have been proposed and compared for both deterministic and stochastic scheduling cases. Mcwilliams et al. (McWilliams *et al.* (2005) ; McWilliams *et al.* (2008)) have studied a truck dock problem for platforms in parcel industries. A genetic algorithm coupled with a simulation model have been applied to minimize total travel time.

Freight transferred in an LTL network comes in different sizes and volumes. Because of the variety of manual systems such as forklifts or pallet jacks used during the transhipment process, the handling approach is labour intensive and costly (Bartholdi et Gue (2000)). In fact, in these terminals, the cost of handling material constitutes the major share of platform operational costs (Gue (1999)). As a result, platform performance relies on more detailed transhipment plans in addition to ordering loading and unloading trucks. In early studies on cross-docks with two doors, the problem was formulated and solved by implementing a heuristic method (Maknoon et Baptiste (2009)), which is an NP-Hard problem (Sadykov (2012)). Although this research provides insight into the structure of a solution, in reality, platforms with multiple doors need to be dealt with.

In a terminal with multiple doors, Alpan *et al.* (2011b) have studied truck scheduling problems that minimize operation costs, which have been expressed as storage and truck replacement costs (truck replacement is a process of temporarily moving semi-unloaded trucks into a parking area in order to liberate a dock). They have also considered a First-In-First-Out (FIFO) transhipment policy. This policy enforces time restrictions on storing products inside the platform. Dynamic programming has been suggested and several heuristics have been proposed to enhance the solution quality (Alpan *et al.* (2011a)).

In this paper, we focus on scheduling transhipment operations in a satellite cross-dock. As mentioned, a satellite terminal has more flexibility in network scheduling. The main priority is to reduce costs, which is possible by investigating more in depth material handling plans.

We consider a real platform with multiple doors. However, as a result of the short processing time, these terminals follow special handling rules. First, the truck replacement is forbidden by platform operational regulations, as it is a costly procedure that may interrupt the guaranteed service (Bartholdi et Gue (2000)). Second, FIFO assumption is not valid in our research. By relaxing this restriction, we will be able to gain further savings in handling costs.

The remainder of this paper is organized as follows : in Section 4.2, we propose a mathematical model to optimize material handling in a satellite cross-dock. The methodology for the solution is discussed in Section 4.3. Computational results are represented in Section 4.4 followed by a conclusion in Section 4.5.

4.2 Problem definition and modeling approach

In this section, we begin by describing the internal transhipment activities in a satellite cross-dock. Then, we formalize the decisions on material handling by introducing a mathematical representation of the problem.

4.2.1 Problem description

In this research, we consider a rectangular cross-dock with multiple doors. A door is assigned to each arriving vehicle by a dock management system. The system follows a First-Come-First-Served (FCFS) policy and truck replacement is forbidden by operational regulations (Bartholdi et Gue (2000)).

Inside the platform, one of three handling decisions needs to be made about an unloaded product : 1) instantly transferring it to the shipping door, 2) moving it to temporary storage, 3) keeping it at the receiving door until the selected outgoing trucks are assigned to the outbound door.

Transferring freight to storage causes double handling. Double handling is a non-beneficial operation that expends platform resources (e.g., opportunity costs of transporters, operators and storage space) (Bartholdi et Gue (2000)). Therefore, in this research, we synchronize truck assignments with material handling decisions to minimize the cost of transhipment. To degrade the problem complexity, we suppose that all trucks have the same capacity and they are either fully loaded or completely unloaded in the platform. In Section 4.4, we relax these restrictions to make the model more compatible with real world problems.

Definition 1. *State ($\Omega_{(i,j)}$) is a time period during which a set of trucks is available at the platform. As soon as one truck (incoming or outgoing) leaves the platform, there will be a transition from one state to another.*

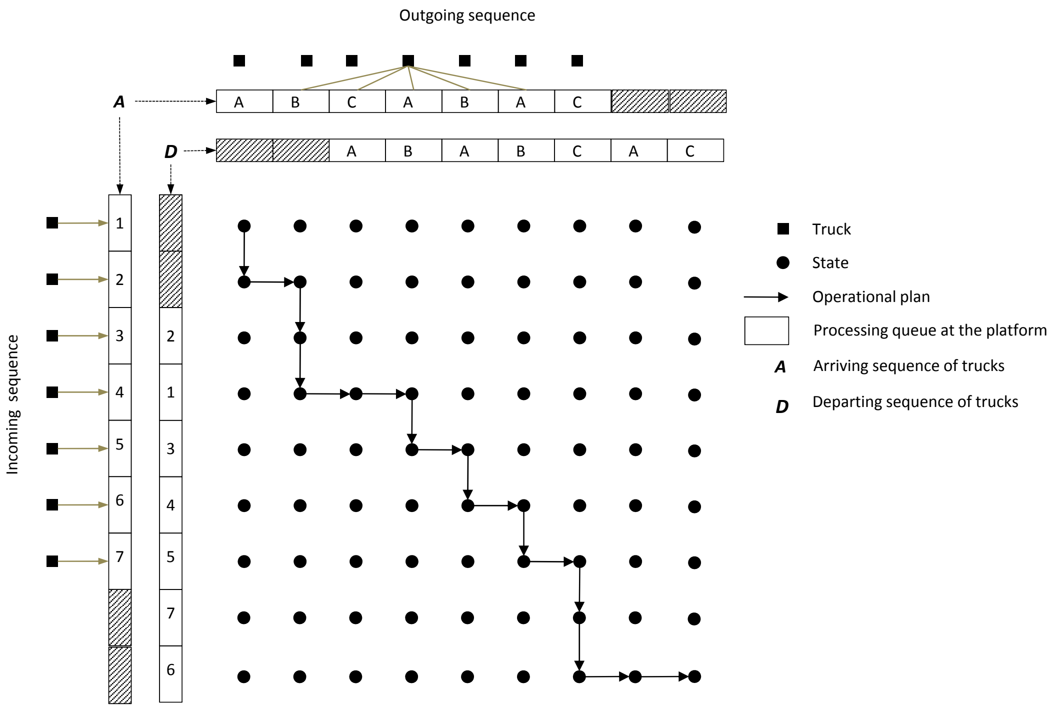

Figure 4.1 Sample solution representation for a platform with two receiving and two shipping docks

Using definition 1, we provide a bounded solution space for the problem. As presented in Figure 1, each node of the graph illustrates a state ($\Omega_{(i,j)}$). Furthermore, we define the operational plan as a path from the first state ($\Omega_{(0,0)}$) to the last one ($\Omega_{(n,n)}$). Based on the aforementioned description, the mathematical model is represented in the following section.

4.2.2 Mathematical model

The material handling problem in a cross-dock terminal is illustrated in Figure 1. In this representation, two indices, ($i \in I$) and ($j \in J$), are the processing queue of receiving ($k^{In} \in K^{In}$) and shipping ($k^{out} \in K^{Out}$) vehicles. The size of the queue is calculated as ($n+g$) in which (n) is the number of incoming/outgoing trucks and (g) represents the number of receiving/shipping docks. The serving order of incoming trucks ($y^{In}_{k^{In},i}$) is known because of FCFS rule. All fleets are homogenous with the same capacity (v). Moreover, each shipping vehicle departs to a specific destination ($d \in D$) and the content of incoming trucks is known (a^d_k). The variables of the model are defined as follows :

Variables :

$Y^{Out}_{k^{In},i}$	Binary variable represents the leaving order of incoming truck ($k^{In} \in K^{In}$)
$P^{In}_{k^{Out},j,d}$	Binary variable represents the arriving order of outgoing truck ($k^{out} \in K^{out}$)
$P^{Out}_{k^{Out},j,d}$	Binary variable represents the leaving order of outgoing trucks ($k^{Out} \in K^{Out}$)
$C^{k^{In}}_{i,j,d}$	Possible direct transfer from an incoming truck k^{In} in state $\Omega_{(i,j)}$ for all trucks departs to destination ($d \in D$)
$O^{k^{Out}}_{i,j,d}$	The number of products directly transfer to outgoing truck ($k^{Out} \in K^{Out}$) for destination ($d \in D$) in state $\Omega_{(i,j)}$
$S_{(i,j),(i',j')}$	Binary variable represents state transition

$$Max \sum_{k^{Out} \in K^{Out}} \sum_{d \in D} \sum_{j \in J} \sum_{i \in I} O^{k^{Out}}_{i,j,d} \tag{4.1}$$

$$\sum_{i \in I} O^{k^{Out}}_{i,j,d} \leq v \left(\sum_{j'=1..j} P^{In}_{k^{Out},j',d} - \sum_{j'=1..j} P^{Out}_{k^{Out},j',d} \right) \qquad \forall j \in J, k^{Out} \in K^{Out}, d \in D \tag{4.2}$$

$$\sum_{k^{Out} \in K^{Out}} O^{k^{Out}}_{i,j,d} \leq \sum_{k^{In} \in K^{In}} C^{k^{In}}_{i,j,d} \quad \forall i \in I, j \in J, d \in D \tag{4.3}$$

$$\sum_{i'=1..i} \sum_{d \in D} \sum_{j \in J} C^{k^{In}}_{i',j,d} \leq a^d_k \left(\sum_{i'=1..i} y^{In}_{k^{In},i'} - \sum_{i'=1..i} Y^{Out}_{k^{In},i'} \right) \qquad \forall i \in I, k^{In} \in K^{In}, d \in D \tag{4.4}$$

$$\sum_{i'=1..i} \sum_{k^{In} \in K^{In}} a^d_{k^{In}} y^{In}_{k^{In},i} - v(\sum_{j'=1..J} \sum_{k^{Out} \in K^{Out}} P^{k^{Out}}_{i,j',d}) \geq M \times (S_{(i,j),(i',j')} - 1) \forall i \in I, j \in J, d \in D \tag{4.5}$$

The objective 4.1 maximizes the number of products that are transferred in just a single handling effort. For each shipping truck, the number of transferred products is constrained by vehicle capacity and the available products on selected states (4.2,4.3). Freight in an arriving truck can be immediately displaced when it is positioned at the receiving door (4.4). In addition, constraint (4.5) ensures that each truck has to leave the platform fully loaded either with products from the receiving door or with products coming from the temporary storage area (M is a big number) .

$$S_{(i-1,j)(i,j)} + S_{(i,j-1)(i,j)} = S_{(i,j)(i,j+1)} + S_{(i,j)(i+1,j)} \quad \forall i \in I, j \in J, d \in D \tag{4.6}$$

$$S_{(0,0)(0,1)} + S_{(0,0)(1,0)} = 1 \tag{4.7}$$

$$\sum_{k^{Out} \in K^{Out}} \sum_{d \in D} O^{k^{Out}}_{i,j,d} + \sum_{k^{In} \in K^{In}} \sum_{d \in D} C^{k^{In}}_{i,j,d} \leq M(S_{(i,j)(i+1,j)} + S_{(i,j)(i,j+1)}) \quad \forall i, j \tag{4.8}$$

$$\sum_{K^{In}} Y^{Out}_{k^{In},i} = 1 \quad \forall i \in \{g, .., n\} \tag{4.9}$$

$$\sum_{d \in D} \sum_{k^{Out} \in K^{Out}} P^{In}_{k^{Out},j,d} = 1 \quad \forall j \in \{1, .., n - g\} \tag{4.10}$$

$$\sum_{d \in D} \sum_{k^{Out} \in K^{Out}} P^{In}_{k^{Out},j,d} = 1 \quad \forall j \in \{g, ..., n\} \tag{4.11}$$

Constraint (4.6) and (4.7) represent the operational plan. In fact, the operational plan links one existing state at a time. There is no direct transfer operation between trucks outside this path (4.8). Finally, constraints (4.9)-(4.11) enforce the sequencing rules for the arrival and departure of incoming and outgoing trucks.

4.3 A sequential priority-based heuristic

To determine an operational plan, three interrelated decisions should be made simultaneously : 1) the processing order of vehicles 2) the unloading order of receiving trucks for each shipping truck 3) displacement strategy (either direct or via storage).

As illustrated in Figure 4.1, we present the operational plan on the graph, with each node indicating a state. The operational plan is a path on this graph from the first state to the last one. Let $\omega^l_{(i,j)} = [T^{K^{In},d}_{(i,j)} | T^{K^{Out},d}_{(i,j)} | \Delta^d_{(i,j)}]$ be one of the decision possibilities ($l \in L$) in state $\Omega_{(i,j)}$. $T^{K^{In},d}_{(i,j)}$, $T^{K^{Out},d}_{(i,j)}$are vectors that represent the number of products in current receiving and shipping vehicles. Vector$\Delta^d_{(i,j)}$ defines the number of products stored inside the platform. The objective value of successor possibility (l') represents a cumulative number of direct transhipments (12) where $H(\omega^l_{(i,j)})$shows the number of products displaced directly in $\omega^l_{(i,j)}$.

$$G(\omega^{l'}_{(i,j)}) = G(\omega^l_{(i,j)}) + H(\omega^l_{(i,j)}) \tag{4.12}$$

This recursive representation has finite number of states. However, the possibilities in each state grow exponentially. The main idea of the proposed approach is to provide a set of rules to restrict the number of possibilities at each state. At the final state $\Omega_{(n,n)}$, the possibility with the highest cumulative value of direct transhipment represents the optimal solution, and the path to reach this possibility is the optimal operational plan. Algorithm 4.3 demonstrates the steps of this approach :

Algorithm 3 Sequential priority-based Heuristic

```
Input :   Known arrival order of incoming trucks
Output :  Processing periods of trucks and operational plan
1 :  Initial assignment
2 :  repeat until reach the last state $\Omega_{(i,j)}$
3 :      for the possibilities $(l)$ in selected state $(i,j)$ do
4 :          $H(\omega^{l}_{(i,j)})$ ← Apply handling decision on $\omega_{(i,j)}$
5 :          $G(\omega^{l'}_{(i,j)})$ ←Calculate the cumulative value of direct transshipment
6 :          $\Omega_{(i',j')}$ ← Choosing next state
```

4.3.1 Initial assignment

This algorithm starts with an initial state in which a set of incoming and outgoing trucks is assigned to the dock door. At the receiving dock, we select incoming trucks based on First-Come-First-Served (FCFS) rule, whereas at the shipping door we restrict the outbound assignment to at most one truck per destination. Therefore, the choices are limited to (D) possible destinations. Finally, we calculate the value of direct transhipments for each destination and choose destinations with the highest value.

4.3.2 Handling decisions

A handling task at possibilities $\omega^{l}_{(i,j)}$ illustrates decisions about product transhipment within the docks. Suppose that $T^{K^{In},d}_{(i,j)}$ and $T^{K^{Out},d}_{(i,j)}$ represent current trucks at the terminal. For each truck at the outbound door, there are some available carriers at the terminals from which goods can be transferred directly. The optimal decision relies on the proper order of selecting carriers as well as the number of products transferred from each vehicle.

Proposition 5. *The knowledge about the order, based on which incoming trucks leave, enables us to set up a priority list for truck selection for decision alternatives.*

If the order of leaving trucks is known, this property provides us with the priority list for incoming trucks. However, in our case, the order is unknown, which makes truck selection a permutation problem. By using a heuristic method, we set up a priority list to select incoming trucks. The highest priority is assigned to an inbound truck with the least amount of remaining products. Then, for each pair of trucks, we displace all of the remaining products to the receiving vehicle destined for a shipping truck. However, the amount of transferred freight is limited to the remaining capacity of an outgoing vehicle. This process is represented in Algorithm 4.3.2.

Algorithm 4 Handling policy

```
Input :    Set of available truck at $\omega^l_{(i,j)}$
Output :   Direct Transhipment
1 :  $PL_{\omega_{(i,j)}} \leftarrow$ Sort incoming trucks available at $\omega^l_{(i,j)}$ based on remaining products
2 :  for all shipping trucks at $\omega^l_{(i,j)}$ do
3 :      for all incoming trucks ordered in priority list($PL_{\omega_{(i,j)}}$) do
4 :          $H(\omega^l_{i,j}) += $ Transfer products from $T^{K^{out},d}_{(i,j)}$ to $T^{K^{out},d}_{(i,j)}$ subject to remaining
             capacity at $T^{K^{Out},d}_{(i,j)}$
```

4.3.3 Choosing the next state

After applying handling decisions, there is no possibility of direct transhipment inside the terminal. Thus, the heuristic starts to liberate one dock for the upcoming state. This process is performed in two steps : first, choosing the leaving truck, second, selecting a replacement truck.

1 Choosing leaving truck

For a set of available trucks at the terminal door, the heuristic approach uses the following priority rules to select the liberating door :

Priority i : Either an incoming vehicle is completely unloaded or an outgoing truck is fully loaded. In this situation, replacing the truck does not violate the optimal solution ; hence, we decide it should leave the platform.

Priority ii : There is a set of outgoing trucks for which this inequality is valid ($T^{K^{out},d}_{(i,j)} + \Delta^d_{(i,j)} \geq v$). In other words, we can employ temporary stored products to dispatch a truck. In this situation, we calculate a “regret” value for each candidate. This value equals the potential increments in the objective function resulting from the next incoming truck in a queue to an outbound candidate. The truck with minimum regret value is then selected.

Priority iii : If selection priorities (i) and (ii) are not successful, the departing truck is chosen from the receiving door with minimum remaining content.

2 Choosing receiving truck

The main idea of truck replacement is similar to the initialization process. The main difference is that here we allow more than one outbound truck to be loaded for the same destination. Similarly, we evaluate the potential number of direct transhipments for each destination. The one with the highest gain is chosen as a candidate. Also, for the incoming candidate, we follow the FCFS queuing policy.

4.4 Computational experiments and discussion

Our heuristic algorithm was implemented in C++ and executed on a computer with a 2.2 GHz CPU and 8 GB of RAM. In addition, we used IBM CPLEX 12.3 as a solver for the mixed integer linear programming model. The time limit has been set to 24 hours. The numerical results are represented in the following subsections. First, we show our data set. Then, we analyze the performance of the algorithm based on different working scenarios. Finally, a framework is suggested in order to apply this algorithm to problems in the real world.

4.4.1 Experimental data

To evaluate the performance of the algorithm, we have considered two platform settings : the first setting consists of 4 docks with 16 trucks at the yard and the second one has 10 docks with 120 trucks. In both settings, the number of receiving and shipping docks is equal, along with the number of trucks.

The results of Tables 4.2 and 4.3 were obtained by using 15 scenarios that were defined based on two platform settings. There are two parameters in each scenario that impact the cost of material handling : 1) the size of serving destinations covered by a platform 2) the distribution of the products inside the incoming trucks. As illustrated in Table 4.1, an average portion of each incoming truck has products for a specific destination. For example, $d_{1:2}^{37.5\%}, d_{3:4}^{12.5\%}$ indicate that inside each arriving truck, 2 destinations have 75% of the products ($2 \times 37.5\%$) while 25% of the contents in each truck carry products to the remaining destinations (Destination 3 and 4).

In all cases, we have randomly generated 5 instances. In all instances, the capacity of the truck is 100 products and we have randomly generated an FCFS queue.

Table 4.1 Data Characteristics

Platform Setting		**# Dest**	**Distribution of Incoming Trucks (DIT)**		
#Truck	*#Doors*		*#1*	*#2*	*#3*
16	**4**	*4*	$d_{1:4}^{25\%}$	$d_{1}^{37.5\%}, d_{2:3}^{25\%}, d_{4}^{12.5\%}$	$d_{1:2}^{37.5\%}, d_{3:4}^{12.5\%}$
		6	$d_{1:2}^{25\%}, d_{3:6}^{12.5\%}$	$d_{1}^{37.5\%}, d_{2:6}^{12.5\%}$	
		8	$d_{1:8}^{12.5\%}$		
120	**10**	*10*	$d_{1:10}^{10\%}$	$d_{1:5}^{13\%}, d_{6:10}^{6\%}$	$d_{1:5}^{15\%}, d_{6:10}^{5\%}$
		15	$d_{1:15}^{6.6\%}$	$d_{1:10}^{5\%}, d_{11:15}^{10\%}$	$d_{1:2}^{15\%}, d_{3}^{10\%}, d_{4:15}^{5\%}$
		20	$d_{1:20}^{5\%}$	$d_{1:10}^{6\%}, d_{11:20}^{3.3\%}$	$d_{1:5}^{10\%}, d_{6:20}^{3.3\%}$

Table 4.2 Comparison of results (setting with 4 doors and 16 trucks)

		Instance														
		1			**2**			**3**			**4**			**5**		
# Dest	**DIT #**	**Best (%)**	**Heur (%)**	**GAP (%)**	**Best (%)**	**Heur (%)**	**GAP (%)**	**Best (%)**	**Heur (%)**	**GAP (%)**	**Best (%)**	**Heur (%)**	**GAP (%)**	**Best (%)**	**Heur (%)**	**GAP (%)**
4	1	89.25	82.5	6.75	87.88	81.63	6.25	88.25	82	6.25	86.63	81.88	4.75	88	83.5	4.5
	2	88.13	82.25	5.88	89.13	82.5	6.63	87.13	82.5	4.63	90	84.63	5.38	87.63	81.75	5.88
	3	87.75	81.24	6.51	87.38	82.88	4.5	88	82.42	5.58	88.88	82.3	6.58	87.75	81.88	5.88
6	1	67.5	63.63	3.88	66.5	62.63	3.88	68	63.38	4.63	69	63.25	5.75	66.5	63	3.5
	2	66.75	62.38	4.38	67.63	63.5	4.13	66.88	63.25	3.63	66.75	63.13	3.63	67.88	63.38	4.5
8	1	47.13	43.75	3.38	47.5	45	2.5	47.63	43.75	3.88	46.75	44.25	2.5	47.75	43.88	3.88
AVG		73.41	74.42	69.30	5.13	74.34	69.69	4.65	74.32	69.55	4.77	74.67	69.91	4.77	74.26	69.57

4.4.2 Analysis of the results

Computational experiments are summarized in Tables 4.2 and 4.3. The column headings are generated instances. In each instance, we report the average direct transhipment ratio, which is calculated as
100 × (Direct transshipment/Total Number of transferred products).

In addition, Table 4.3 shows the comparison between the optimal solution with the one obtained by our heuristic. The columns labeled “Best” report the best percentage of direct transhipments ; a similar value by using a heuristic is shown in “Heur”. Columns labeled “Gap” show the solution gap between two approaches. In addition, we have tested the heuristic approach for the second cross-dock setting, which is presented in Table 4.3.

As shown in Table 4.3, the solution gap is relatively small in our test instances (less than 5% on average). It tends to become smaller as the number of serving destinations becomes larger. This is because there are fewer permutations when selecting the departing trucks, since the number of destination increases. Moreover, the algorithm demonstrates stable solutions by varying the distribution of arriving products.

Table 4.3 and Figure 4.2 denote the computational time and the direct transhipment ratio in all instances for the second platform settings. In all instances, the heuristic method demonstrates efficient and consistent performance. The average ratio of direct transhipment considerably depends on the ratio between the number of platform doors and the number of destinations. As shown in Figure 4.2, for the same ratio, the heuristic algorithm has shown almost the same transhipment ratio. Moreover, the average CPU time is less than 5 seconds, which makes it a good choice for real world problems.

We have measured the performance of material handling by external factors that are imposed by network scheduling : the amount of serving destinations and the distribution of incoming products. As illustrated in both tables, with the same platform settings the material handling costs increase when the amount of assigned destinations is higher. For example, in cases with 4 destinations, fewer than 20% of products need double handling, whereas for 8 destinations, double handling is required for more than 50% of the products. This issue should be considered when designing the platform layout. On the other hand, the cost of material handling is less sensitive to the distribution of products inside the arriving trucks.

In the previous section, we assumed a homogenous fleet with the same capacity in which all trucks have to be fully loaded. In practical cases, this restriction is rarely achieved. The presented heuristic can be applied to these problems with small modifications. Moreover, this method can be easily adapted in order to cope with uncertainties about the order of arrival of trucks.

Table 4.3 Results of the heuristic algorithm (platform setting with 10 doors and 120 trucks)

		Instance						
# Dest	**DIT#**	1	2	3	4	5	AVG	Time(Second)
	1	86.88	86.88	86.43	85.3	87.18	86.54	3.2
10	2	72.2	72.7	71.4	71.85	72.77	72.18	4.7
	3	81.83	83	82.13	78	82.93	81.58	3.7
	1	64.4	66.95	62.52	69.37	63.63	65.37	3.8
15	2	54.07	54.98	54.62	54.68	54.92	54.65	5.5
	3	68.6	69.67	69.45	65.22	71.53	68.89	5.1
	1	53.92	48.87	52.98	52.22	48.42	51.28	4.6
20	2	50.72	52.08	51.52	54.02	51.5	51.97	5.8
	3	54.33	54.38	53.9	54.47	54.12	54.24	6.3

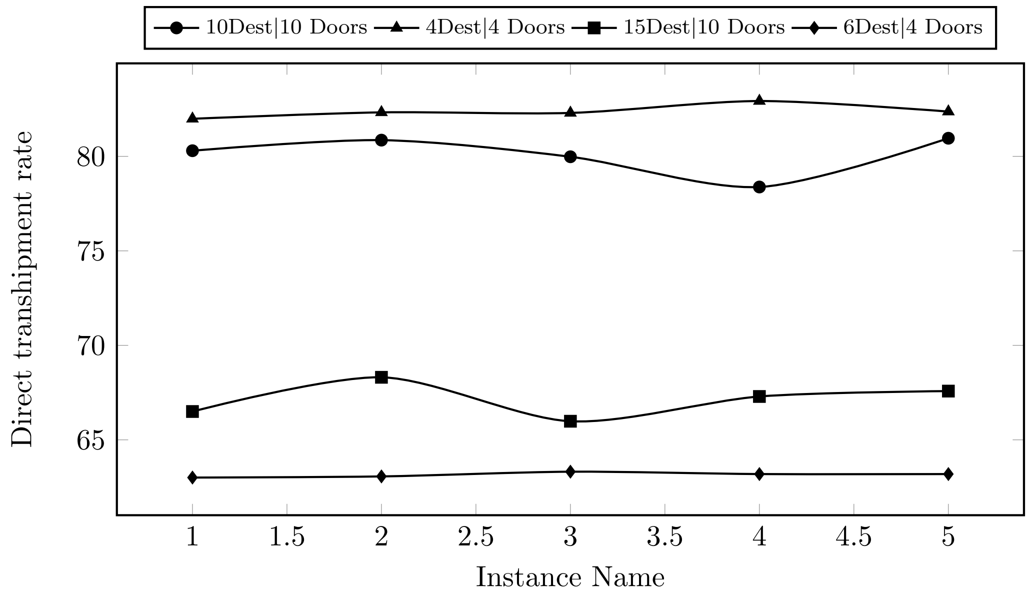

Figure 4.2 Comparing the average direct transhipment for the second data settings

4.5 Conclusion

In this paper, we have studied a scheduling problem in a satellite cross-dock. Because of the restrictions on truck scheduling in an LTL network, these terminals have more flexibility with processing trucks. Thus, their efficiency is based on the cost of internal transhipment.

We have modeled the internal handling process and identified double handling as a key factor in transhipment deficiencies.

A mathematical model is represented to optimize their internal processes. Moreover, a sequential priority-based heuristic is represented, which can be applied to real world problems. The computational results demonstrate efficient performance of this algorithm. Also, we have tested the heuristics for different scenarios. Based on the experiments, the distribution of products inside the arriving trucks has a negligible impact on the cost of material handling. However, the amount of destinations served that is assigned to the platform can dramatically increase the handling cost, which should be considered as a factor when the cross-dock layout is designed.

This work can be combined with studies in dock door assignment (see Gue (1999)) as a concurrent scheduling approach that simultaneously optimizes the operational plan and transfer distance.

CHAPTER 5

ARTICLE 4 : SCHEDULING INTERNAL TRANSHIPMENT IN A MULTIPLE DOOR CROSS-DOCK : A VARIABLE NEIGHBORHOOD SEARCH APPROACH

Chapter Information : An article based on this chapter was submitted for publication. M.Y.Maknoon, F.Soumis, and P.Baptiste.Scheduling internal transhipment in a multiple door cross-dock : A variable neighborhood search approach.

Abstract The process of internal transhipment in cross-docking terminals is a complicated task that directly affects the operational cost of the platform. To facilitate an efficient process, a proper scheduling system is required that provides explicit decisions about the transfer of products. In this paper, we introduce a scheduling model to optimize the internal transhipment process. The model synchronizes the processing periods of trucks at inbound and outbound doors with decisions about product displacement. We formalize the problem and present a mathematical formulation that provides an optimal solution for small instances. In addition, we present several search operators that are incorporated in a variable neighborhood search framework to tackle real life problems. Computational results depict the performance of this algorithm. Finally, we analyze different strategies for managing internal transhipment based on situations encountered on the platform and demonstrate economical savings in the transhipment process.

keyword Cross-docking, Material handling, Scheduling, Integer programming, Variable neighborhood search

5.1 Introduction

The profit margin of transportation and logistics companies is not high enough to ignore the existence of cross-dock. Cross-dock is a consolidation point that realises savings in freight transportation and inventory cost (Waller *et al.* (2006)). Compared with warehousing, cross-docking succeeds in excluding the two costly operations of storage and retrieval (Bartholdi et Gue (2004)). Despite these remarkable advantages, cross-docking is beneficial as long as its operational costs do not overwhelm its economical advantages (Bartholdi et Gue (2000)).

The aim of this paper is to study the operational activities inside the cross-docking terminal. The platform is characterized by the mode of internal transporters. It employs forklifts or pallet jacks to shift freight. Moreover, it operates in a pre-distributional mode (Yan et Tang (2009)). That is, the supplier determines the final destination of received freight, and the platform does not have any additional duties other than transferring and consolidating products. Examples of these platforms can be found in less-than-truckload and retail industries.

The problem of material handling deals with decisions about how to transfer freight inside the cross-dock. Product transhipment is a costly process, as it interlocks with platform resources (e.g., workforces, transporter devices). Therefore, the manner of carrying out transhipment duties has an impact on the platform's operational cost. As the amount of resources is limited, one way to decrease the operational expense is to avoid excessive work during the consolidation procedure.

Considering the aforementioned concern, product double handling can be identified as a source of deficiencies. Double handling is a demanding task that influences important cross-dock performance indicators such as handling rate, output rate, workforce requirement, processing time and requirement of internal storage (Schwind (1995)). However, with an appropriate scheduling system, it is possible to directly ship 50-100% of receiving freights via cross-dock (ZINN (1994)).

In this study, we represent a scheduling model for the purpose of eliminating the excessive displacement of products inside the platform. The model synchronizes the processing periods of vehicles with decisions about internal transhipment and provides an explicit operational plan for the cross-dock.

The rest of the paper is organized as follows : Section 5.2 presents a review of scheduling problems that arise in a cross-docking terminal. In Section 5.3, a mathematical representation that is implemented by a general solver and obtains optimal solutions for small instances is provided. Moreover, in Section 5.4, several search operators incorporated in a variable neighborhood search framework to deal with real life problems are presented. The experimental results and a discussion are detailed in Section 5.5, followed by the conclusion in Section 5.6.

5.2 Literature Review

The scheduling problem at the cross-docking terminal has been vastly studied in recent years (see Van Belle *et al.* (2012) for an extensive review). These studies are categorized in two main groups based on the platform size.

The first group of studies considers a conceptual cross-dock with single receiving and shipping doors. The scheduling model determines an order of truck processing to minimize operational time. The problem was first introduced by Yu et Egbelu (2008), in which authors proposed a mixed integer programming model and developed a heuristic method as a resolution approach. Further various meta-heuristic algorithms have been proposed that consider different issues and limitations on transhipment procedures (e.g., limitations on the platform storage capacity, uncertainties on the arrival time of incoming trucks). (Boloori Arabani *et al.* (2011) ;Soltani et Sadjadi (2010) ;Larbi *et al.* (2011)). Although these studies provide good insight about the problem structure, they are not applicable to a platform with multiple doors.

The second group of studies has investigated internal operations in multiple door cross-docks. For known truck schedules at outbound doors,Mc-Williams and coworkers(McWilliams *et al.* (2005) ;McWilliams *et al.* (2008)) have studied the problem of material handling in parcel hub terminals. These centers are equipped with automated conveyors and sortation systems in which all products are transferred directly from inbound to outbound doors without being stored inside the platform. The problem presented suggested decisions about the unloading order of vehicles at the inbound doors to reduce transfer time span.

Liao *et al.* (2012) and Alpan *et al.* (2011b) have studied the scheduling problem in a multiple door cross-dock where temporary storage is permitted. The scheduling model synchronizes internal transhipment with truck processing order. In (Liao *et al.* (2012)) authors have considered a known schedule of trucks at outbound doors. The scheduling decision regulates unloading sequences of vehicles at each receiving door to minimize total weighted tardiness. Six meta-heuristics have been developed and compared as resolution approaches.

Conversely, for known sequence of trucks at receiving doors, Alpan *et al.* (2011b) studied the scheduling problem to reduce operational cost. They have expressed operational costs as product storage and truck pre-emption costs. They have also considered a First-In-First-Out (FIFO) transhipment policy that is applied in practice. This policy enforces time restrictions on storing products inside the platform. A dynamic programming approach has been suggested and several heuristics have been proposed to enhance the solution quality (Alpan *et al.* (2011a)).

Our study is different from most previous studies (except Alpan *et al.* (2011b) ;Liao *et al.* (2012)) in that we consider a platform with multiple doors and allow the products to be temporarily stored inside the platform.

The difference between our model and the model presented by (Alpan *et al.* (2011b) and Liao *et al.* (2012) is that it focuses on truck sequencing at both inbound and outbound doors.

Our main contributions are : (i) presenting a mathematical formulation of the problem that succeeds to provide the optimal solution with commercial software (ii), developing a variable neighborhood search heuristic that provides a good solution for practical problems (iii), presenting and comparing different strategies on managing platform operations and demonstrating savings in the cost of material handling.

5.3 Problem description and mathematical representation

In most cases, cross-docks have a rectangular shape with multiple doors around it. The inbound doors are assigned to receiving trucks. Shipping vehicles are loaded at the outbound doors. Inside the platform, manual handling systems (e.g., forklifts or a pallet jack) are employed for internal transhipment. For an unloaded product, if the assigned shipping truck is located at the door, it directly displaces to outgoing vehicle. Otherwise, it moves to a temporary storage area to be marshaled for future shipments. This latter decision causes product double handling.

Definition : for a given arrival and departure order of trucks at inbound and outbound doors, the loading and unloading plan is a set of decisions to transfer products within doors. The best plan is the one that has the least amount of double handling of products.

With the aforementioned definition, the scheduling decision determines the arrival and departure order of trucks at inbound and outbound doors that results in a minimum number of products being double handled in the loading and unloading plan.

In this model, we assume that all trucks have the same capacity. They are available at the beginning of the planning horizon and are either fully loaded or unloaded at the platform. Moreover, we do not consider any restrictions on the platform's internal storage.

Table 5.1 Summary of notations

Sets :	
t	Set of seqeuncing position ($t \in T$)
k	Set of incoming trucks ($k \in K$)
d	Set of outgoing destinations ($d \in D$)
Indices :	
In	Arriving trucks
Out	Departing trucks
l	Indices represents transhipment relation at sequence t between a pair of trucks at inbound and outbound doors. Value 1 demonstrates the case in which the outgoing truck starts loading at sequence t when the incoming one is presented at the platform. Value 0 Otherwise.
Parameters :	
$\mathbf{a_k^d}$	Amount of products inside the incoming truck ($k \in K$) for each destination ($d \in D$)
g	Number of receiving or shipping doors
$\mathbf{s_t^{In}}$	Parameter that regulates the sequencing rules for arriving order of incoming trucks
$\mathbf{s_t^{Out}}$	Parameter that regulates the sequencing rules for the departing order of incoming trucks
v	Truck capacity
m	Big number
Variables :	
$\mathbf{Y_{k,t}^{In}}$	Binary variable represetnts the arriving order of incoming truck ($k \in K$) at sequence ($t \in T$)
$\mathbf{Y_{k,t}^{Out}}$	Binary variable represents the leaving order of incoming truck ($k \in K$) at sequence ($t \in T$)
$\mathbf{O_{t,t'}^d}$	An arc represets an outgoing truck, ship to destination ($d \in D$), arrive at the platfrom in sequence ($t \in T$) and leave it in sequnece ($t' \in T$).
$\mathbf{X_{k,t}^{d,l}}$	variable in $[0,1]$ which represents the portion of products that directly transferred from incoming truck ($k \in K$) to destination ($d \in D$) in sequence ($t \in T$)

Figure 5.4.2(a) illustrates a schema of the proposed model. The loading and unloading plan is represented by two sets of sequences. Sequence ”In” represents the starting period of unloading incoming trucks, whereas sequence ”Out” demonstrates its departing period.

Parameters s_t^{In} and s_t^{Out} regulate sequencing rules for incoming trucks. s_t^{In} is equal to 1 when unloading a new truck is not permitted (positions 20 and 21 in Figure 5.4.2(a)). A similar rule applies to the leaving order, which is enforced by s_t^{Out} (positions 2 and 3 in the schema presented).

For incoming vehicles ($k \in K$), variables $Y_{k,t}^{In}$ and $Y_{k,t}^{Out}$demonstrate their arrival and departure orders. For example, in Figure 5.4.2(a), trucks IX arrives at position 2 and departs the platform at position 4.

The arcs represent the processing period of outgoing trucks ($O_{t,t'}^d$). This represents that the outgoing truck starts loading for destination ($d \in D$), in sequence ($t \in T$), and leaves the platform in ($t' \in T$).

Incoming vehicles contain products for different destinations (a_k^d) and each shipping vehicle departs to one destination ($d \in D$). Finally, variable $X_{k,t}^{d,l}$ represents the transhipment decision. For each sequence ($t \in T$), it demonstrates the porion of products in truck ($k \in K$) that are directly transferred to ougoing vehicles loaded to destination ($d \in D$). The notations are summarized in Table 5.1, based on what the scheduling model is then :

$$\mathcal{Z} = Max \quad \sum_{l \in \{0,1\}} \sum_{k \in K} \sum_{t \in T} \sum_{d \in D} a_k^d x_{k,t}^{d,l} \tag{5.1}$$

$$\sum_{t \in T} \sum_{d \in D} O_{0,t}^d = g \tag{5.2}$$

$$\sum_{j \in T | j > t} O_{t,j}^d + \sum_{k \in K} Y_{k,t}^{Out} + s_t^{Out} = 1 \quad \forall t \in T \tag{5.3}$$

$$\sum_{d \in D} \sum_{i \in t | i < t} O_{i,t}^d = \sum_{d \in D} \sum_{j \in T | j > t} O_{t,j}^d \quad \forall t \in T \tag{5.4}$$

$$\sum_{i \in T | i < t} O_{i,t}^d + \sum_{k \in K} Y_{k,t}^{In} + s_t^{In} = 1 \quad \forall t \in T \tag{5.5}$$

$$X_{k,t}^{d,0} \leq O_{i,j}^d \quad \forall i, j, t \in T | i < t, j > t \tag{5.6}$$

$$X_{k,t}^{d,1} \leq O_{t,j}^d \quad \forall t \in T, j \in T | j > t \tag{5.7}$$

$$\sum_{l \in \{0,1\}} X_{k,t}^{d,l} \leq \sum_{i \in T | i < t} (Y_{k,i}^{In} - Y_{k,i}^{Out}) \quad \forall t \in T, \forall d \in D, \forall k \in K \tag{5.8}$$

$$\sum_{l \in \{0,1\}} \sum_{t \in T} X_{k,t}^{d,l} \leq 1 \quad \forall k \in K, \forall d \in D \tag{5.9}$$

$$\sum_{k \in K} \sum_{i \in T | i < t} a_k^d Y_{k,i}^{In} \geq v \times (\sum_{i \in T | i < t} \sum_{j \in T | j = i+1, j \leq t} O_{i,j}^d) \quad \forall t \in T, d \in D \tag{5.10}$$

$$\sum_{l \in \{0,1\}} \sum_{t \in T} \sum_{k \in K} a_k^d X_{k,t}^{d,l} \leq v(O_{i,j}^d) + m(1 - O_{i,j}^d) \quad \forall d \in D \quad \forall i, j \in T | j > i \tag{5.11}$$

The objective function (5.1) maximizes the total number of products that are transferred directly from inbound to outbound doors. Constraints (5.2), (5.3), (5.4) and (5.5) regulate truck sequencing rules. In the first sequencing position, (g) outgoing trucks, parameter (g) represents the number of shipping doors, are positioned at platform door (5.2). For the illustrated example in 5.4.2(a), we assign two outgoing trucks in position 1. Then, in the next (g) sequences, the incoming trucks are assigned to platform dock (no truck leaves the platform) (5.3), which are positions 2 and 3 in the illustrative example. Thereafter, the replacements can be either at inbound or outbound doors. Finally, in the last (g+1) sequences all the vehicles leave the platform (5.5) (positions 20-22 in Figure 5.4.2(a)).

There is a possibility of direct transshipment from receiving vehicle ($k \in K$) to truck loaded for destination ($d \in D$), if both are presented at the platform (5.6), (5.7) and (5.8). Constraint (5.9) makes sure that the value of direct transhipment does not exceed the content of inbound truck. The full truckload assumption is represented in (5.10). Finally, constraint (5.11) ensures that the amount of transferred products does not exceed the capacity of outgoing vehicles.

5.4 Variable neighborhood search approach

Variable Neighborhood Search (VNS) is a framework that follows the idea of systematically exploring the different neighborhood structures of the problem instances. The extensive introduction of this approach can be found in Hansen et Mladenović (2003).

In our VNS implementation (Algorithm 5), the initial solution is obtained by a heuristic method based on FIFO policy. Then, a set of operators is systematically launched in a sub-routine to search for local optima. Whenever an operator detects an improved solution, the sub-routine restarts from the first operator. Otherwise, a perturbation operator is used to perturb the current solution. This procedure terminates when no improvement is found after a finite number of attempts (φ^{Max}).

Algorithm 5 General Schema of VNS algorithm

```
Output : The best sequences that minimize double handling
1 :  $S_1 \leftarrow$ Initial solution ; $\varphi \leftarrow 1$
2 :  while $\varphi < \varphi^{Max}$ do
3 :      $S_2 \leftarrow$ Best solution after applying a series of search operators on $(S_1)$
4 :      if $S_2 > S_1$ then $S_1 \leftarrow S_2$ ; $\varphi \leftarrow 1$
5 :      else $\varphi + +$
6 :      $S_1 \leftarrow Perturb(S_2, \varphi)$
```

5.4.1 Initial solution based on FIFO policy

For a random arrival order of incoming trucks, the First-In-First-Out (FIFO) transhipment policy is used to rapidly construct the departing order of outgoing trucks and loading and unloading plan. The goal of this method is to simulate the type of decisions taken in practice (Alpan *et al.* (2011b)).

To preserve the feasibility of the solution, we only build the departing succession of trucks, and we duplicate it for the arriving orders. First, we randomly assign incoming trucks to k positions of departing sequences successively. Then we start from the first position and examine the cumulative flow of receiving freight for each destination. As soon as sufficient products are available for one destination in sequence (t'), we insert an outgoing vehicle shipped to destination ($d \in D$) and move forward all of the remaining incoming vehicles. The course of action repeats until the leaving order of all outgoing vehicles is determined. Figure 5.4.2(a) illustrates an initial solution for our problem.

5.4.2 Search operators

In this section, we introduce operators that are embedded in a search sub-routine. As presented in the previous section, the scheduling problem has five sequencing variables. Four sequencing variables are related to the arrival and departure order of the trucks. The fifth one is the loading and unloading plan.

For simplicity in representation, each operator accompanies a quadruple (IA,ID,OA,OD) in which IA and ID are the arrival and departure order of inbound trucks. Similarly, OA and OD represent the arrival and departure order of outbound vehicles. We underline each order if it is returned by the operator.

Let $Pos^{In}(w) = t_1$ and $Pos^{Out}(w) = t_2$ represent arriving $t_1 \in T$ and departing $t_2 \in T$ position of truck (w). The number of incoming vehicles that are unloaded and have left platform before position t_1 is denoted by $PreI(t_1)$. $PreO(t_1)$ marks the number of outgoing trucks that leave the platform ahead of t_1. Based on the above definitions, the operators are described in the following parts.

O1-Ordering operator (IA, ID, OA, OD)

In the ordering operator, the loading and unloading plan are of greatest importance. It applies a search mechanism to detect, for a known processing order of trucks, the optimal plan that minimizes the cost of double handling. In other words, it looks to determine the best relation between the processing periods of the trucks at inbound and outbound doors.

Let n be a node that presents current receiving and shipping trucks at the sequence position t which holds the following data :

- $T^{In}_{k,t}$, $T^{Out}_{d,t}$: Vector representing the content of trucks at inbound and outbound doors
- $\Delta_{d,t}$: Vector representing the quantity of products stored inside the platform
- B_t : An upper bound value of the objective function in position ($t \in T$)
- C_t : Objective function showing the cumulative number of direct transshipment until position ($t \in T$)

As presented in algorithm 6, the search method starts from an initial node in which all trucks are assigned to platform doors. Then, it carries out the process of freight transshipment and updates the objective value. For each outgoing truck, it selects an incoming vehicle to transfer products. The selection criterion is based on their departing order. Thereafter, it moves to the next position with two possibilities. The first node is created by replacing a vehicle at inbound doors. All products remained in the truck are moved to the storage area, upper bound is updated and the truck is replaced. The second node is generated by changing trucks at outbound doors. In this case, if the shipping truck is partially loaded, the stored products are used to fill the truck. The algorithm terminates when all trucks leave the cross-dock and the node with the highest objective value is recognized as the optimal solution. Figure 5.4.2(b) illustrates the loading and unloading plan after applying ordering operator on 5.4.2(a).

Algorithm 6 Ordering operator (IA, ID, OA, OD)

```
Input : Known arriving and departing order of trucks at incoming and outgoing doors
Output : Optimal loading and unloading plan
1 :  Initialize; list← create an initial node
2 :  While list $\neq \phi$
3 :      $n \leftarrow$ Select a node from the list
4 :      Apply transhipment process on node $n$
5 :      Discard the node $n$ if it is redundant
6 :      Select the replacement truck from outbound doors and create a node
7 :      Select the replacement truck from inbound doors and create a node
```

During the search process, a node is discarded when its upper bound is inferior to the best-known solution. Additional nodes are discarded based on the property, described as follows :

Property 4. *For two nodes n and $n^{'}$ at position $t = t^{'} \in T$ in which $PreI(t) = PreI(t^{'})$ and $PreO(t) = PreO(t^{'})$. If the remaining content of the trucks is similar, the node that has a better upper bound and higher objective value dominates the other one.*

Démonstration. The condition indicates that two different loading and unloading plan results in duplicating the situation at the platform. Therefore, we choose the plan that has had a better outcome up until this position. □

O2- Relocating operator (<u>IA</u>, <u>ID</u>, <u>OA</u>, <u>OD</u>)

The relocating operator selects two trucks and exchanges their processing periods. Variations are (<u>IA</u>, <u>ID</u>, OA, OD) and (IA, ID, <u>OA</u>, <u>OD</u>). Let $Pos^{In}(w) = t_1$ and $Pos^{Out}(w) = t_2$ represent the processing period of truck w ; similarly, $Pos^{In}(w') = t_3$ and $Pos^{Out}(w') = t_4$ are the arriving and departing order of truck $w^{'}$. After applying this operator, we will have :$Pos^{In}(w) = t_3$,$Pos^{Out}(w) = t_4$, $Pos^{In}(w') = t_1$ and $Pos^{Out}(w') = t_2$. After each replacement, ordering operator (IA,ID,OA,OD) is called to build a new loading and unloading plan. If the solution is improved, the operator accepts the current arrangement ; otherwise, it switches back to the previous arrangement. For the sample instance, represented in Figure 5.4.2(a), we choose trucks I and IX, and replace their positions, which are illustrated in Figure 5.4.2(c), then the ordering operator is applied. The result is presented in 5.4.2(d).

O3-Vicinity - Ordering operator ($\underline{\text{IA}}$, ID, $\underline{\text{OA}}$, $\underline{\text{OD}}$)

At the same time, the vicinity-ordering operator examines the repositioning of trucks' arrival or departure orders with the loading and unloading plan. The operator has three variants ($\underline{\text{IA}}$,ID,OA,OD), (IA,ID,$\underline{\text{OA}}$,OD) and (IA,ID,OA,$\underline{\text{OD}}$). The idea behind this operator is to capture the decrement in the cost of double handling by partially varying the truck processing period. We apply the same search structure described in Algorithm 6. However, in steps 6 and 7, the operator generates more nodes, as it has to examine different scenarios concerning the arriving or departing trucks at each position ($t \in T$).To overcome the computational burden, we restrict the truck selection to those that are chosen by the following criteria :

– For the variation ($\underline{\text{IA}}$, ID, OA, OD), the algorithm searches for the arriving order of incoming trucks in addition to the loading and unloading plan. In this case, we examine the leaving order of incoming vehicles (ID). All trucks that are not assigned until the sequence position (t) and will leave the platform in the next α order are selected.
– In (IA, ID, $\underline{\text{OA}}$, OD) the algorithm computes, for each destination, the amount of remaining products in inbound trucks. All destinations that have the above average remaining are considered.
– To select the departing candidates in (IA, ID, OA, $\underline{\text{OD}}$), the algorithm inspects all the outbound trucks that could be dispatched to their final destination by using products stored inside the platform. Among them, first we consider the truck that has the least remaining capacity. Second, we also consider other outbound vehicles in which more than β% of their capacity is loaded.
  Figure 5.4.2(e) illustrates the loading and unloading plan after applying the vicinity-ordering operator on the arrival order of incoming trucks ($\underline{\text{IA}}$, ID, OA, OD) for the instance shown in Figure 5.4.2(a).

O4-Block-shift Operator ($\underline{\text{IA}}$, $\underline{\text{ID}}$, $\underline{\text{OA}}$, $\underline{\text{ID}}$)

The block-shift operator is applied only to the sequences of inbound trucks. A block is defined as a set of adjacent replacements of incoming trucks in a loading and unloading plan. For instance, in Figure 5.4.2(a), there are four blocks of inbound vehicles ({IX, VI, VIII}, {III}, {VII, IV, V, II},{I,X}).

All the blocks are selected as candidates and the operator starts from the first block and shifts it forward. After each replacement, we use vicinity-ordering operator ($\underline{\text{IA}}$, ID, OA, OD) to build a new arrangement for the loading and unloading plan. Figure 5.4.2(f) represents the results of using a block shift operator on Figure5.4.2(a). Here, the operator replaces {IX, VI, VIII} after {VII, IV, V, II} and executes a vicinity-ordering operator.

O5 - Exchanging Operator (IA, ID, OA, OD)

The exchanging operator randomly selects γ trucks and shuffles their departure positions. This operator has two variants (IA, ID, OA, OD) and (IA, ID, OA, OD). This means the selected vehicles are either from an inbound or outbound area. The newly constructed arrangement may violate the feasibility condition ; therefore, we apply vicinity-ordering operator (IA, ID, OA, OD) or (IA, ID, OA, OD) after each repositioning. In Figure 5.4.2(a), the operator selects the position of 4 trucks : VI, II, III, X and shuffle their positions (Figure 5.4.2(g)) ; after the vicinity-ordering operator is executed, the results are illustrated in Figure 5.4.2(h).

O6- Perturbation Operator (S, φ)

The perturbation operator is applied to escape from the local optima. In our implementation, we apply a perturbation operator to the processing periods of incoming vehicles (IA, IL, OA, OL).

To do that, first we perturb the departing order. The procedure is to perform a series of individual swaps. Let $Pos^{Out}(w) = t_1$ and $Pos^{Out}(w') = t_2$ are two randomly selected incoming trucks, the swap move is defined by applying $Pos^{Out}(w) = t_2$, $Pos^{Out}(w') = t_1$. The amount of swap procedures increases by the number of unsuccessful attempts in the decent step (φ).

Second, we randomly re-assign the arrival order. In order to have a feasible solution, we start from the first leaving truck and randomly assign the arriving order to one of the available positions before the selected one. The process continues until all of the incoming orders are assigned.

5.5 Computational experiments :

The VNS heuristic was coded in C++ and tested on a computer with 2.2 GHz CPU and 8 GB of RAM. IBM CPLEX 12.3 was chosen as a solver for the mixed integer linear programming model. The computational results are presented in three sub-sections. First, we explain the calibrated values of the search parameters and the order of applying operators in our implementation. Second, we describe the test instances and provide a comparison between the solution computed by VNS and the one reported from the mathematical model. Third, we analyze the savings in the cost of material handling under different scheduling restrictions.

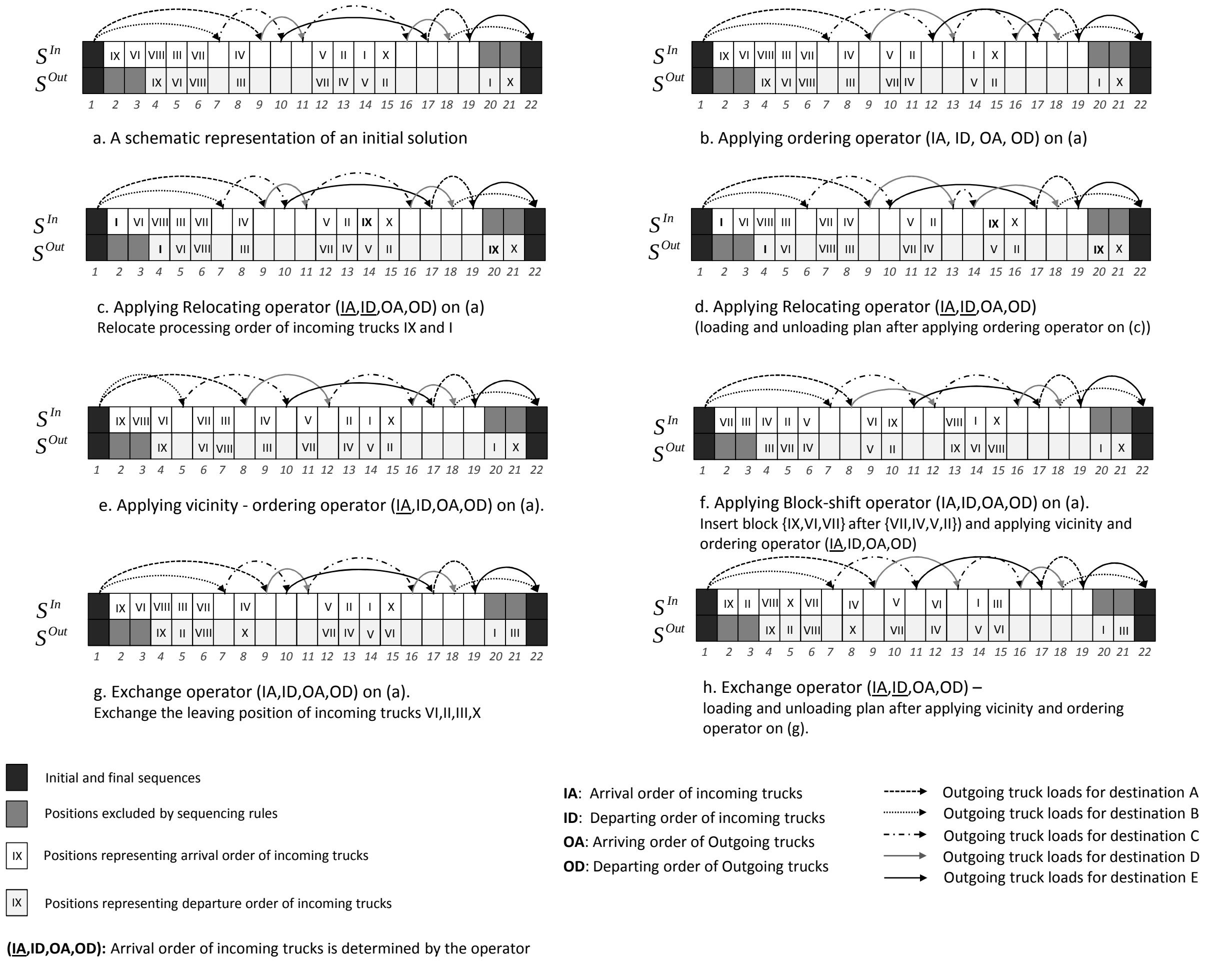
a. A schematic representation of an initial solution
b. Applying ordering operator (IA, ID, OA, OD) on (a)
c. Applying Relocating operator (IA,ID,OA,OD) on (a)
Relocate processing order of incoming trucks IX and I
d. Applying Relocating operator (IA,ID,OA,OD)
(loading and unloading plan after applying ordering operator on (c))
e. Applying vicinity - ordering operator (IA,ID,OA,OD) on (a).
f. Applying Block-shift operator (IA,ID,OA,OD) on (a).
Insert block {IX,VI,VII} after {VII,IV,V,II}) and applying vicinity and ordering operator (IA,ID,OA,OD)
g. Exchange operator (IA,ID,OA,OD) on (a).
Exchange the leaving position of incoming trucks VI,II,III,X
h. Exchange operator (IA,ID,OA,OD) –
loading and unloading plan after applying vicinity and ordering operator on (g).
Initial and final sequences
Positions excluded by sequencing rules
Positions representing arrival order of incoming trucks
Positions representing departure order of incoming trucks
(IA,ID,OA,OD): Arrival order of incoming trucks is determined by the operator
IA: Arrival order of incoming trucks
ID: Departing order of incoming trucks
OA: Arriving order of Outgoing trucks
OD: Departing order of Outgoing trucks
Outgoing truck loads for destination A
Outgoing truck loads for destination B
Outgoing truck loads for destination C
Outgoing truck loads for destination D
Outgoing truck loads for destination E

Figure 5.1 Model Schema (Platform with 2 receiving and 2 shipping doors)

5.5.1 Tuning parameters :

Table 5.2 represents the order of launching operators incorporated in search sub-routing. In our implementation, each variation of operators is executed separately and their execution order is based on intensification and diversification aspects of the search. Moreover, Table 5.3 reports the calibrated values of search parameters. These values are tuned based on the instance characteristics. Finally, we choose the first improvement strategy. That is, as soon as a better solution is found, we accept the improvement and restart the search sub-routine.

Table 5.2 Order of operators used in search sub-routine

1 :	Ordering operator (IA, IL, OA, OL)	**2 :**	Relocating operator ($\underline{\text{IA}}$, $\underline{\text{IL}}$, OA, OL)
3 :	Relocating operator (IA, IL, $\underline{\text{OA}}$, $\underline{\text{OL}}$)	**4 :**	Vicinity - Ordering operator ($\underline{\text{IA}}$, IL, OA, OL)
5 :	Vicinity - Ordering operator (IA, IL, $\underline{\text{OA}}$, OL)	**6 :**	Vicinity - Ordering operator (IA, IL, OA, $\underline{\text{OL}}$)
7 :	Exchanging Operator ($\underline{\text{IA}}$, $\underline{\text{IL}}$, OA, OL)	**8 :**	Exchanging Operator (IA, IL, $\underline{\text{OA}}$, $\underline{\text{OL}}$)
9 :	Block-shift Operator ($\underline{\text{IA}}$, $\underline{\text{IL}}$, OA, OL)		

Table 5.3 Calibrated value of search parameters

Parameter	**Description**	**Calibrated value**
$\varphi^{\mathbf{Max}}$	Termination criteria	If the number of unsuccessful attempt reach 1.2×*(number of serving trucks)*
α	Number of look ahead positions in vicinity-ordering operator ($\underline{\text{IA}}$,ID,OA,OD)	Number of receiving doors (g)
β	The loading percentage of outgoing trucks used in vicinity-ordering operator (IA,ID,OA,$\underline{\text{OD}}$)	80% of vehicle capacity (v)
γ	Number of vehicles selected in exchange operator	For ($\underline{\text{IA}}$,$\underline{\text{ID}}$,OA,OD), a random value between 30 to 70 percent of serving trucks (k). For (IA,ID,$\underline{\text{OA}}$,$\underline{\text{OD}}$), a value equal to the number of destinations(d).

5.5.2 Experimental settings and computational results

To test our algorithm, 48 instances were generated. They are characterized by four main parameters : the quantity of cross-dock doors (4 to 8) equally divided between inbound outbound area, the amount of processed trucks (16 to 64) evenly employed as an incoming and outgoing trucks, the amount of shipping destinations (4 to 12) and the distribution of products inside the incoming trucks (B and U). For each instance, we randomly generate two product distribution themes named (B) and (U). In (B) all of the outgoing destinations have relatively equal demand, whereas in (U) some destinations have considerably more demand than others.

Table 5.4 summarizes our computational experiments. The first five columns describe instances and their characteristics. The columns are : instance number, number of trucks, number of doors, number of destinations and product distribution. Columns under "VNS" report the computational results of VNS heuristics. Because of the stochastic nature of the VNS method, it was performed 5 times for each instance. The column "Rate" shows the average amount products that are directly transferred. The rate is calculated as 100× *(total amount of products directly transferred / total number of processed products)*. Columns labeled "μ" and "σ" report the average and standard deviation of VNS execution time (CPU time in second).

Column "Math. Rate" reports the best solution obtained by the mathematical model. We allow 24 hours running time for CPLEX. Out of 48 instances, we are able to solve 12 of them to optimality, which are marked as a bold value. For the rest of the instances, we report the best-known solution.

Finally, Column "Diff." represents the difference between the rate of direct transshipment obtained by the mathematical model and VNS approach. The negative value indicates that the VNS approach outperforms the solution of the mathematical model.

In all of the instances, the VNS heuristic exhibits good behavior. On average, the gap between the VNS algorithm and the optimal one is 1.38%. For the rest of the instances, we can see that the resulting solution of the VNS algorithm is mostly better than the solution that we obtain by using the mathematical model, and most of the time the direct transhipment rate obtained by the VNS method is 5% higher. In some cases, this gap exceeds 20%.

Regarding the CPU time, on average our VNS approach was executed in three minutes. For most of the instances, the execution time is less than a minute ; however, for some instances (e.g. 45,47) its run time increased to 20 minutes. This implies that in some cases, finding an improvement is challenging during the search process and we could find a better solution after applying a considerable perturbation on the current solution.

Overall, the proposed search algorithm demonstrates stability for finding a scheduling plan, which could be used in practical problems.

Table 5.4 Computational results

Instance characteristics					**VNS**				
No.	# Truck	# Door	# Dest	Dis.	Rate (%)	Time[1] μ	Time[1] σ	**Math. Rate (%)**	**Diff. (%)**
1	16	4	4	B	88.5	0.06	0.01	**88.75**	0.25
2	16	4	4	U	88.5	0.08	0	**90.5**	2
3	16	4	5	B	76.6	0.14	0.03	**77.25**	0.65
4	16	4	5	U	78.7	0.11	0.02	**79.25**	0.55
5	16	4	6	B	66.45	0.12	0.03	**68.5**	2.05
6	16	4	6	U	64.75	0.1	0.01	**67.25**	2.5
7	20	4	4	B	88.36	0.39	0.12	**90.6**	2.24
8	20	4	4	U	89.48	0.27	0.1	**89.8**	0.32
9	20	4	5	B	79.96	0.3	0.01	**81**	1.04
10	20	4	5	U	78.44	0.34	0.12	**79.4**	0.96
11	20	4	6	B	70.56	0.36	0.05	**72.6**	2.04
12	20	6	8	U	67.88	0.72	0.29	**69.8**	1.92
13	32	8	8	B	85.95	7.64	2.46	87.38	1.43
14	32	8	8	U	90.93	24.65	10.73	88.63	-2.31
15	32	8	10	B	77.15	29.75	2.52	76.25	-0.9
16	32	8	10	U	78.58	22.4	6.51	78.75	0.17
17	32	8	12	B	66.08	10.42	1.23	67.5	1.42
18	32	8	12	U	65.88	21.15	6.66	67.88	1.99
19	40	4	4	B	92.88	5.69	1.83	87.8	-5.08
20	40	4	4	U	91.02	6.47	1.36	87.9	-3.12
21	40	4	5	B	80.34	12	2.04	82.5	2.16
22	40	4	5	U	84.18	11.71	5.17	82.3	-1.88
23	40	4	6	B	74.48	44.61	4.74	77.3	2.82
24	40	4	6	U	74.76	18.93	3.47	76.6	1.84
25	40	6	6	B	90.74	21.94	8.54	89	-1.74
26	40	6	6	U	91.08	9.45	2.68	83.8	-7.28
27	40	6	7	B	82.8	35.75	8.78	83.7	0.9
28	40	6	7	U	83.8	18.78	4.49	80.4	-3.4
29	40	6	8	B	76.4	61.69	17.45	78.1	1.7
30	40	6	8	U	82.4	38.14	11	80.3	-2.1
31	60	4	4	B	92.85	58.07	12.18	84.07	-8.78
32	60	4	4	U	93.16	45.62	11.69	85.47	-7.69
33	60	4	5	B	83.05	101.29	55.77	78.07	-4.98
34	60	4	5	U	83.43	88.78	8.68	77.27	-6.16
35	60	4	6	B	70.83	497.58	62.71	68.93	-1.9
36	60	4	6	U	73.01	220.65	12.04	68.87	-4.14
37	60	6	6	B	92.35	67.81	17.1	78.53	-13.82
38	60	6	6	U	92.92	70.66	6.75	76.67	-16.25
39	60	6	7	B	84.65	279.88	53.39	75.47	-9.18
40	60	6	7	U	85.33	179.28	36.75	67.33	-18
41	60	6	8	B	77.84	719.32	34.45	62.47	-15.37
42	60	6	8	U	79.05	392.03	36.22	68.73	-10.32
43	64	8	8	B	91.5	395.44	86.92	45.75	-45.75
44	64	8	8	U	91.38	426.96	17.59	78.06	-13.32
45	64	8	10	B	80.18	1700.25	240.3	69.44	-10.74
46	64	8	10	U	82.83	1106	163.18	62.75	-20.08
47	64	8	12	B	69.44	1302.25	203.2	39.5	-29.94
48	64	8	12	U	82.49	1108.98	138.54	62.63	-19.87
Average					**81.54**	**190.94**	**27.08**	**76.27**	

[1] CPU time in second

5.5.3 Strategies for managing platform operations

Another interesting aspect of this problem is that it could be used to evaluate savings in the cost of material handling under different strategies to manage platform operations. To this end, we have investigated three scenarios that are based on flexibilities in scheduling trucks. Moreover, we use a FIFO based heuristic method to simulate the types of decisions that are taken in practice. These scenarios are :

Basic Scenario : The platform has a restricted order of processing inbound and outbound trucks. The cross-dock has to process trucks based on the first-come-first-served rule (Bartholdi et Gue (2000)). The decision is to build a loading and unloading plan based on the FIFO transhipment policy. We choose the initial heuristic to provide a solution in this scenario.

First Scenario : As with the base scenario, we duplicate orders obtained by the FIFO transhipment policy. However, we apply the ordering operator to rearrange the loading and unloading plan.

Second Scenario : The platform has flexibility in scheduling outbound trucks. The decision is to synchronize the outgoing order of trucks with the internal transhipments. Here, we use all of the variety of the proposed search operators that deal with scheduling outbound trucks. Each operator is applied only once and their order is the same as the one that we use in our VNS implementation. The difference between this scenario and the basic scenario is to simultaneously optimize the outgoing order of the truck with internal transhipment instead of using the FIFO policy.

Third scenario : The platform has the flexibility to determine the processing order of trucks at inbound and outbound doors. The proposed VNS algorithm is employed in this scenario.

To have a fair comparison between the cases described, we use the solution of the base scenarios as an initial solution for other scenarios ($s = 1, 2, 3$). Table 5.5 reports the percentage differences between the solution values of scenarios 1, 2 and 3 with the base scenarios. The value calculated as *(Solution of scenario (s) - Solution of base scenario)/100.*

The results are reported based on the five time executions. The column named ”First Sc.” reports the average and standard deviation between the computed value by the ordering operator and the heuristic solution. After applying the ordering operator, one can notice that for most of the small instances, the solution of applying the FIFO transhipment policy is as good as the optimal one. However, as the amount of processing trucks increases, there is a considerable gap between the proposed loading and unloading plan by the FIFO policy and the optimal one. In some instances, this value exceeds 5%.

Columns under ”Second Sc.” report the results of applying search operators to improve the outbound processing period of trucks as well as internal transhipment. One can remark, for the fixed processing order of incoming trucks, that there was an 8% improvement in the cost of material handling.

Finally, Columns under ”Third Sc.” represent the improvement in the cost of material handling, if the platform has freedom to process receiving and shipping trucks. This time we are able to obtain, on average, 13% improvement in the cost of material handling compared to our basic scenario.

To conclude, in reality the cross-docking platform has some limitations on processing trucks, which is a result of network planning and priorities. Here, we have analyzed three strategies to manage the internal transhipment based on the degree of freedom in processing trucks. We show that for all scenarios, optimizing the transhipment process results in a 3% to 13% improvement in the cost of internal transhipment.

Table 5.5 Marginal improvement in handling rate

	First Sc.		**Second Sc.**		**Third Sc.**	
#	μ(%)	σ(%)	μ(%)	σ(%)	μ(%)	σ(%)
1	4.65	0.57	9.7	3.31	20.06	0
2	0	0	0	0	12.69	0.38
3	0	0	9.78	1.55	12.08	3.17
4	0	0	3.54	1.98	10.1	1.72
5	0	0	0	0	2.14	0.34
6	0	0	5.56	3.03	10.81	2.59
7	0	0	4.31	2.26	9.52	0.5
8	0	0	0.99	1.06	6.53	0.87
9	2.1	0.31	4.1	1.02	8.36	0.72
10	1.8	0.2	3.9	0.98	7.42	0.81
11	7.11	1.16	11.21	2.41	17.8	3.7
12	0	0	5.49	4.83	11.73	1.1
13	2.68	1.03	11.93	2.73	13.23	2.84
14	0	0	7.58	2.91	10.83	1.7
15	4.14	1.2	5.97	1.89	9.17	1.94
16	0	0	3.86	1.70	8.18	1
17	0	0	1.95	2.82	6.85	2.88
18	0	0	5.58	6.73	13.28	0.39
19	4.22	0.63	11.24	2.04	15.85	2.67
20	2.65	0.99	5.14	1	9.62	2.31
21	8.13	1.82	16.06	2.2	18.2	2.69
22	5.86	2.24	10.1	2.09	16.01	1.75
23	3.71	1.62	11.3	1.34	14.67	2.7
24	5.05	1.38	12.07	4.49	15.44	4.97
25	1.06	1.18	4.5	1.81	7.45	1.87
26	0.52	0.24	11.21	2.2	15.28	1.29
27	4.61	1.13	14.68	2.32	17.56	2.65
28	3.14	1.12	10.11	1.21	14.51	2.71
29	3.11	1.5	9.06	2.04	13.59	1.84
30	1.7	1.35	6.85	1.43	10.29	3.27
31	3.68	1.26	8.49	1.45	10.57	2.32
32	4.21	1.62	9.23	1.01	11.51	2.34
33	9.52	0.92	13.04	1.89	18.24	1.64
34	5.96	1.81	12.02	1.94	15.79	2.12
35	11.33	1.43	14.42	2.19	18.41	1.97
36	5.83	1.84	11.66	2.04	15.34	3.48
37	2.77	1.22	6.04	3.31	9.31	1.55
38	2.73	0.53	9.84	0.84	12.04	1.67
39	3.71	2.07	8.52	3.05	11.21	3.1
40	2.05	0.54	11.63	2.26	14.14	2.26
41	8.7	1.79	16.09	2.37	18.6	2.43
42	4.59	0.56	12.51	1.14	16.51	1.11
43	3.66	1.77	13.57	2.81	14.39	2.78
44	2.29	1.04	8.91	0.55	11.2	0.81
45	5.52	1.74	14.58	3.17	16.79	2.39
46	2.43	0.86	10.49	1.01	12.8	2.39
47	6.72	1.96	13.41	0.87	16.09	1.33
48	2.28	0.18	7.76	1.47	10.99	2.26
Average	**3.21**		**8.75**		**12.77**	

5.6 Conclusion

The operational cost of cross-docking platforms is highly dependent on the process of internal transshipment. In this paper, we represent a scheduling problem to synchronize the processing order of trucks with internal transshipment. We represent a mathematical model of the problem, which is able to solve small instances. In addition, we develop a variable neighborhood search heuristic to tackle the real problems. Several search operators are presented. The computational results depict the good performance of the algorithm in terms of solution quality and execution time. Moreover, we have analyzed different strategies for managing the transhipment process that the platform faces in reality. We have adopted our proposed method for each case. The results demonstrate considerable savings in the cost of material handling compared to the policies that we use in practice. The models developed and approaches are designed when a platform operates in a deterministic environment ; however, in reality there is uncertainty about the order of arriving trucks, which could provide a new direction for future research.

CHAPTER 6

GENERAL DISCUSSION

In this dissertation, we have studied the scheduling problem in cross-docking platforms. The model studied integrates the loading and unloading process of trucks with internal transshipment. Moreover, we have introduced important elements of each scheduling model. The elements are :decisions to determine an order for processing trucks at the terminal, allocating trucks to the platform door, and managing internal transshipment. The platform studied employs manual handling systems (e.g., forklifts) for internal transshipment. In these centers, handling freight is costly, as it is in direct contact with platform key resources such as operators and transporters. The objective of the problem is to minimize the excessive handling operation inside the platform.

6.1 Summary of results

In Chapters 2 and 3, we have proposed models and algorithms to schedule material handling for the platform with a single receiving and shipping door.

First, we have proposed a dynamic programming model to schedule internal transshipment when the processing order of trucks is known. We have also embedded the dynamic programming model as an optimizer in a stochastic search framework.

Second, we have presented a mixed integer linear programming formulation of the problem. We have proposed a set of valid inequalities embedded in a branch and bound framework to optimally solve the problem. The key idea behind the branch and bound method is to avoid symmetries occurred by branching on the path linking the inbound and outbound sequences. The most challenging part of the problem relies on determining the arrival order of trucks. If the processing order of an incoming truck is known a priori, then determining the optimal solution would be less complicated. The proposed model has adaptation capabilities with restrictions imposed by network planning. For example, we can add different constraints to impose some restrictions on processing trucks, or add some limitations on the storage space. Moreover, we have proposed a diving heuristic to find the initial solution. Based on the computational results, this method can find the near optimal solution in a reasonable time.

Chapters 4 and 5 have examined the scheduling problem for a cross-dock with multiple doors. In the platform setting with single receiving and shipping doors, the order of trucks is equal to their processing periods. However, in a multiple door cross-dock, the processing periods of trucks is different from their serving order. Therefore, in order to represent the processing period of the truck we use two types of sequencing variables. One sequence is used for the arrival order and the other one demonstrates the leaving order. Chapter 4 studies the scheduling model in a case where there are restrictions on processing incoming trucks. An example of this scheduling model is at a satellite cross-dock, which is responsible for local deliveries. We have presented a sequential priority based heuristic algorithm to tackle the problem. The main idea in this method is to provide transferring rules for the search algorithm. One of the advantages of this approach is its application in real life problems. For example, it is possible to partially load trucks during the consolidation process (relaxing the full truckload condition).

Finally, Chapter 5 has examined the general scheduling problem for a multiple door platform. We have presented a new formulation, which has a considerably lower number of constraints and variables. Moreover, we have developed a variable neighborhood search heuristic. The challenging part of this algorithm is to compute the value of an objective function. Here, we have designed a search mechanism to efficiently compute the value of objective function. We have developed several search operators to broadly explore the solution space. The results have revealed that this approach could find a near optimal solution. In addition, for other problems, the operational plan obtained outperforms ones that were obtained from the mathematical model.

6.2 Problem assumptions and transformations

As discussed in the previous section, in this dissertation we have developed several models and algorithms to tackle the problem of double handling inside the platform. These algorithms were proposed under certain assumptions about physical and operational characteristics of the cross-dock. In the following section, we present some of these assumptions and provide guidelines for adopting the models and algorithms presented in this dissertation.

6.2.1 Reducing the traveling distance of products within door

The objective of the scheduling models presented in this dissertation is to minimize the excessive product handling inside the platform. However, as stated in Chapter 2, product travel distance within doors has an impact on the platform's operational costs. This problem is studied under the name of dock-door assignment.

In the dock-door assignment problem, first it is assumed there is complete information about the processing queue of a truck at each platform dock, and second, the amount of products that transfer within doors is known. Then, the problem is to determine the assignment of each queue of trucks to the platform dock in order to minimize total product transfer distance.

Based on the aforementioned descriptions, the optimal loading and unloading plan constructed to minimize the double handling of products can be supplied for the dock-door assignment problem in order to reduce the total transferring distance. The procedure is :

1. Construct the processing queue of trucks at each platform door.
2. Determine the amount of products that passes within doors.
3. Use one of the algorithms proposed by (Bartholdi et Gue (2000) and Tsui et Chang (1992)) to optimize products' traveling distance

Figure 6.2.3 represents these steps

6.2.2 Variation on the amount of inbound and outbound doors

In Chapter 4 and 5, we assumed that the platform doors are equally distributed between the inbound and outbound area. However, all of the presented approaches can be used for the different distribution of doors between inbound and outbound areas.

In this case, we should mention that if the amount of doors is equal to the number of outgoing destinations, then all of the products are directly transferred. But, this will cause inefficiencies on processing incoming trucks, as fewer doors are available for them.

6.2.3 Floating dock

Another subject of interest is the concept of floating dock (Peck (1983)). In most research, it is assumed that each door is capable of processing either an incoming or an outgoing truck. The reason behind this assumption relies on the practical limitation. Each type of door (inbound or outbound) requires a certain type of equipment to process trucks.

The concept of a floating dock is introduced by Peck (1983). These types of doors are capable of processing both incoming and outgoing trucks at the same time. In this situation, we can modify the mathematical model presented in Chapter 6 for this case. As the total number of platform doors is a constant value (g), we can use the following modifications in model (6.1)-(6.11) :

We eliminate constraints (6.2) and (6.4) and add the following constraint instead :

$$\sum_{i=1}^{t}\sum_{k\in K} Y_{ki}^{In} - \sum_{1=1}^{t}\sum_{k\in K} Y_{ki}^{Out} + \sum_{\forall i<t}\sum_{\forall j>t}\sum_{d} O_{ij}^{d} \leq g \qquad \forall t \tag{6.1}$$

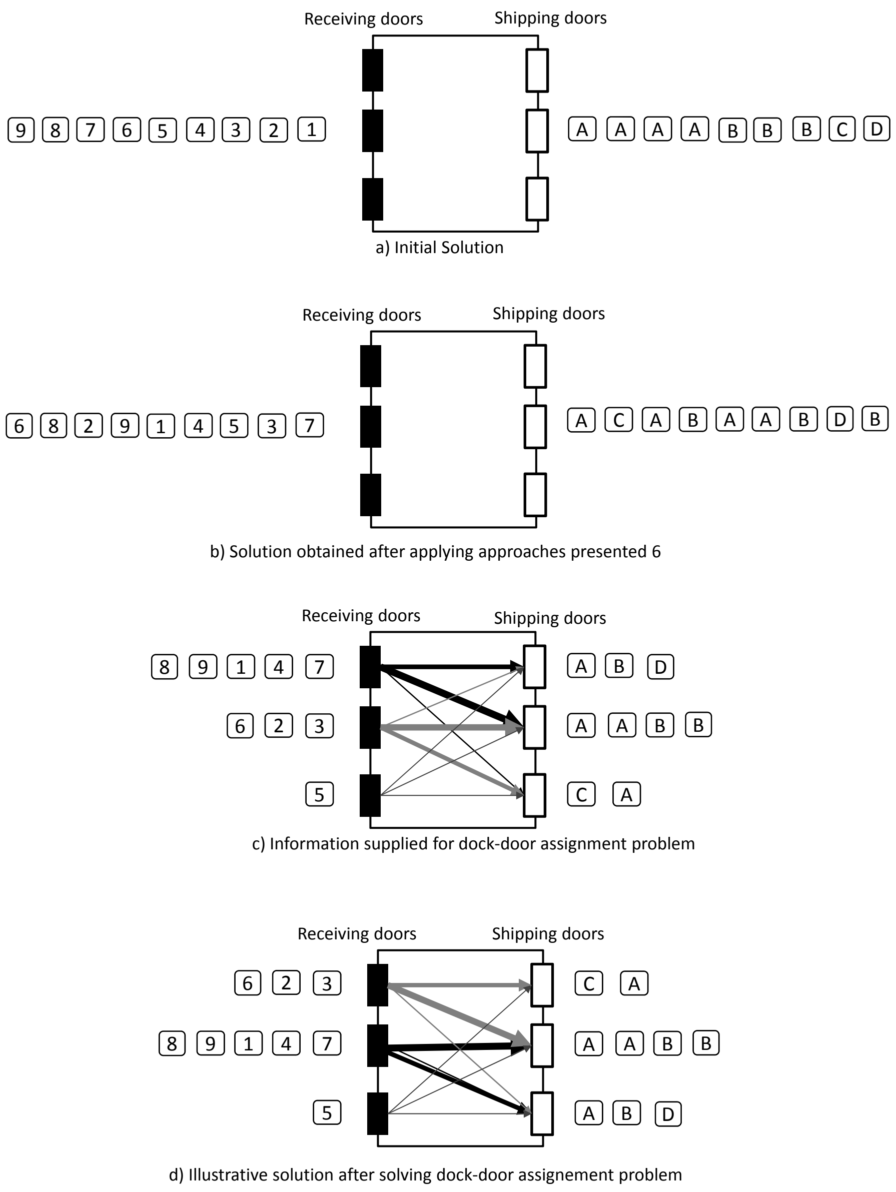

Figure 6.1 Transformation to dock-door assignment problem

6.2.4 Limitation on platform internal storage

Generally, in practice the LTL platform has sufficient internal space for processing trucks. In this situation, restricting the internal storage capacity does not have an effect on the solution procedure. If there are limitations, the following changes can be made to the models presented in Chapters 3, 4, 5 to cope with this situation :

1. For the model presented in Chapter 3, constraint (7.2) makes sure the storage capacity

is not violated. In this constraint, SC represents the storage capacity.

$$\sum_{i'=1..i-1} a^{k,d} X_{i,o}^{k,d} - v \sum_{i'=1..i-1} \sum_{o \in O} F_{(i',o)(i',o+1)}^{d} \leq SC \quad \forall i \in I, d \in D \tag{6.2}$$

2. For models (5.1)-(5.11) and (6.1)-(6-11), it is required to add a bounded variable representing the amount of products stored inside the platform at each state.

All of the algorithms can also be extended to take into account the limitation on platform storage capacity. However, considering the limitation on platform storage brings additional combinatorial aspects to the problem. This aspect deals with the decision about transferring products stored inside the platform. Furthermore, limiting the storage capacity may cause an infeasible solution. One way to avoid infeasibility is to allow truck pre-emption during the consolidation procedure (Alpan *et al.* (2011b)).

Remark : Minimizing the excessive handling movement directly leads to a reduction in the amount of products stored inside the platform, while this result does not necessarily respect the limitation on the platform storage space.

6.2.5 Relation to operational time

In general, cross-docking platforms may require respecting some time restrictions imposed by the logistics network (e.g. cross-dock operates in courier industries). In this section, we demonstrate an approach to translate the loading and unloading plan based on the time indicators and the way to relate it to the total operational time.

According to the literature, there are two definitions that consider platform operational time :

In the first definition, total processing time is defined as a summation of each product's transfer time. The transfer time includes : time for searching, picking up, transferring and dropping off the products. If we assume a single unit of time for each transfer attempt, then minimizing double handling leads to a reduction in total processing time. This assumption provides a good approximation, as most of the time spent in a product's transhipment is related to searching, picking up and dropping off (Gue (1999)).

In the second point of view, processing time is defined as the time between unloading the first product until it is time to load the last one in a shift (Makespan). In the literature, authors (Boysen *et al.* (2010)) have considered three assumptions on scheduling problems that minimize the Makespan. First, all trucks are fully loaded or unloaded. Second, they do not consider the internal transferring and assume that products can instantly be loaded in outgoing trucks as soon as they are available at the platform. Third, the unloading and loading operation can be done independently.

Based on these assumptions, they have defined a sequence as a period of time that the truck unloads/loads at the platform and provide an upper and lower bound of processing time. The lower bound is defined if we are capable of simultaneously processing an incoming and an outgoing truck in one sequence. Conversely, the upper bound happens when we process one truck in each sequence.

The first and the second assumptions that have been explained are applicable to our scheduling model. However, we have assumed dependencies between loading and unloading trucks. Therefore, in each sequence, only one truck can leave the platform.

In order to translate our solution in terms of Makespan, we can modify the notation of the path that we named "loading and unloading plan". In Figure 6.2.5(a), we demonstrate a solution obtained by the model expressed in Chapter 3 and 4.

In these models, if we permit the simultaneous departure of trucks at inbound and outbound doors, then we can modify the structure of the path to represent it. Figure 6.2.5(b) illustrates this modification. In this graph, nodes(i,j) is directly connected to node (i+1,j+1) if there exists a path from node (i,j) to (i+1,j) and from (i+1,j) to (i+1,j+1) or a path from (i,j) to (i,j+1) and from (i,j+1) to (i+1,j+1). By applying this method, we can express the loading and unloading plan in terms of processing time. In the example, after applying the aforementioned method, trucks can be processed in 7 sequences.

Remark : Minimizing the operational time stated by the first and second definition does not necessarily minimize the double handling of products, as they do not explicitly consider internal transfer.

Finally, we can employ the aforementioned translation method to impose some restrictions on the truck arrival and departure time in our model.

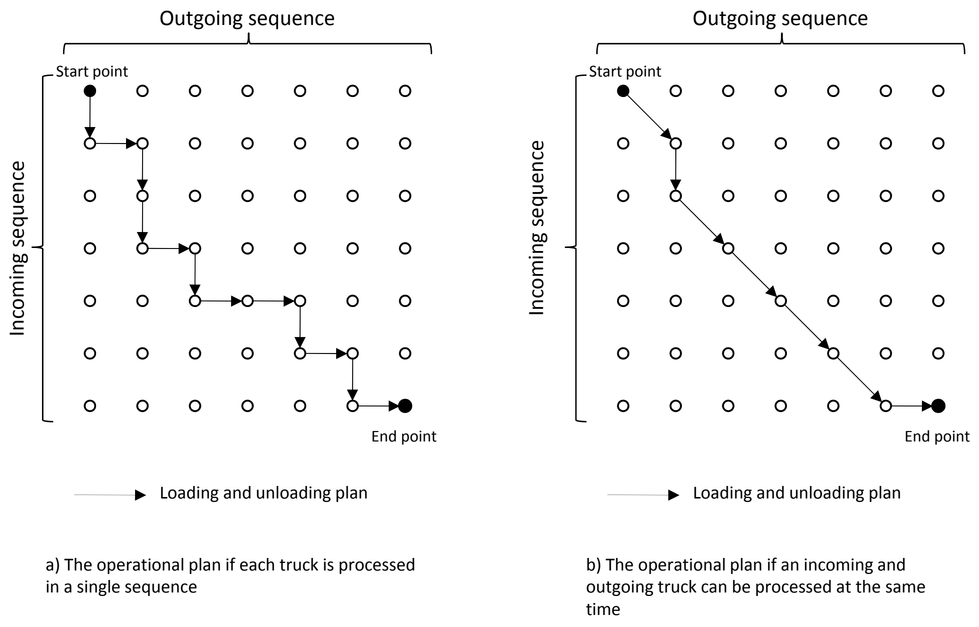
Outgoing sequence
Start point
Incoming sequence
End point
Loading and unloading plan
a) The operational plan if each truck is processed in a single sequence
Outgoing sequence
Start point
Incoming sequence
End point
Loading and unloading plan
b) The operational plan if an incoming and outgoing truck can be processed at the same time

Figure 6.2 Expressing the scheduling solution as a processing time

6.2.6 Full truckload assumptions

In this thesis, we have assumed that each inbound and outbound truck is fully loaded when it arrives or departs the platform. In practice, there are cases when trucks are partially loaded. The full truckload assumption can be relaxed, as the number of arriving and departing trucks and their contents are known in advance.

All models and algorithms presented in this thesis are compatible with a case when incoming trucks are partially loaded.

To relax the full truckload assumption on the outgoing trucks, we consider some artificial products for each destination stored in the platform. These products are available at the beginning of the planning horizon. With this modification, we can then solve the problem with a full truckload assumption.

CONCLUSION

In this dissertation, we have studied the problem of material handling in cross-docking terminals. Our main goal is to develop a scheduling model that is capable of synchronizing the inner transshipment decisions with the loading and unloading orders of trucks. First, this model is investigated for the conceptual cross-dock setting with single receiving and shipping doors. Second, the problem has been studied for a real platform setting with multiple receiving and shipping doors. We have presented models and algorithms as resolution approaches. The following paragraphs outline the main results and discussion about potential research directions.

In chapter 1, we have presented a classification scheme for scheduling problems in cross-dock terminals. In most of the studies reviewed, the platform has an unlimited internal storage capacity. This assumption could be suitable for the small and medium cross-dock. However, in the case of large platforms, the capacity of internal storage could be problematic, something that should be considered.

In chapter 2 and 3, we have formalized the scheduling problem and have proposed a dynamic programming model to schedule internal transshipment when the processing order of trucks is known. The proposed method is integrated in a stochastic search framework to improve the processing order of trucks. In addition, a mixed integer linear programming model is provided. We have introduced some families of valid inequalities and have proposed several structural properties. These properties are embedded in a path-branching algorithm to find the optimal solution. This approach is able to solve instances of up to 40 trucks.

In chapters 4 and 5, we have studied the scheduling problem for the platform with many receiving and shipping doors. First, we have presented a fast heuristic algorithm to find the optimal solution in case there are restrictions on the arrival order of trucks. Second, we have presented a new mathematical model for the general problem in which all sequences have to be determined.

Moreover, we have introduced several search operators that are embedded in a variable neighborhood search to find a good loading and unloading plan. The results show our heuristic is a suitable choice in practice.

In addition, we have also analyzed savings in the cost of material handling. We have studied the effect of two external factors on the handling operation : the amount of destinations served and the distribution of products upon arrival. Based on the experiments, we can state that increasing the number of destinations significantly augments double handling. This has been considered a factor in network scheduling and planning. However, variations in the distribution of arriving products has a negligible impact on the cost of double handling. Moreover, we have investigated the effect of imposing restrictions on the arrival and departure order of trucks on the cost of material handling. Comparing the results based on a first-in-first-out policy, we are able to reduce the cost of material handling even if the platform has no flexibility with processing trucks.

Concerning internal transshipment, in this thesis we have focused on developing scheduling models to minimize excessive product displacement inside the platform and we do not take into account the assignment of trucks to platform doors, which is a quadratic assignment problem. It would be interesting to combine these two problems in future research.

Moreover, a scheduling model is developed for the static environment. We have assumed that all of the trucks are available at the beginning of the planning horizon or that they arrive in a certain order. However, in practice, there is uncertainty about the arrival time of the vehicles. One future direction would be to investigate platform operation in the stochastic environment.

Another aspect of interest would be to develop an online scheduling model. In the thesis, we have developed a finite planning horizon for cross-docking operations. However, in reality some platforms operate 24 hours a day and it is difficult to distinguish a working shift for them. By having uncertainties about the arrival and departure order in an infinite planning horizon, online scheduling models would be a suitable direction for future studies.

REFERENCES

ACKERMAN, K. B. (1997). Cross-docking in the warehouse. *Practical handbook of warehousing*, Springer. 425–432.

ALPAN, G., LADIER, A.-L., LARBI, R. et PENZ, B. (2011a). Heuristic solutions for transshipment problems in a multiple door cross docking warehouse. *Computers & Industrial Engineering*, 61, 402–408.

ALPAN, G., LARBI, R. et PENZ, B. (2011b). A bounded dynamic programming approach to schedule operations in a cross docking platform. *Computers & Industrial Engineering*, 60, 385–396.

APTE, U. M. et VISWANATHAN, S. (2000). Effective cross docking for improving distribution efficiencies. *International Journal of Logistics*, 3, 291–302.

ARABANI, A. B., GHOMI, S. F. et ZANDIEH, M. (2010). A multi-criteria cross-docking scheduling with just-in-time approach. *The International Journal of Advanced Manufacturing Technology*, 49, 741–756.

BAPTISTE P, PENZ B., L. R. (2007). Polynomial time solution methods for some cross-docking problems. *International Conference on Industrial Engineering and Systems Management.*

BARTHOLDI, J. et GUE, K. (2000). Reducing labor costs in an ltl crossdocking terminal. *Operations Research*, 48, 823–832.

BARTHOLDI, J. et GUE, K. (2004). The best shape for a crossdock. *Transportation Science*, 38, 235–244.

BOLOORI ARABANI, A., FATEMI GHOMI, S. et ZANDIEH, M. (2011). Meta-heuristics implementation for scheduling of trucks in a cross-docking system with temporary storage. *Expert systems with Applications*, 38, 1964–1979.

BOTTANI, E., MONTANARI, R. et RIZZI, A. (2004). A fuzzy multi-attribute dynamic approach for optimal management of shipping and receiving flows of a distribution centre. *Proceedings of the logistics research network 2004 annual conference. Dublin, Ireland.*

BOYSEN, N. (2010). Truck scheduling at zero-inventory cross docking terminals. *Computers & Operations Research*, 37, 32–41.

BOYSEN, N. et FLIEDNER, M. (2010). Cross dock scheduling : Classification, literature review and research agenda. *Omega*, 38, 413–422.

BOYSEN, N., FLIEDNER, M. et SCHOLL, A. (2010). Scheduling inbound and outbound trucks at cross docking terminals. *OR spectrum*, 32, 135–161.

CHEN, F. et LEE, C.-Y. (2009). Minimizing the makespan in a two-machine cross-docking flow shop problem. *European Journal of Operational Research*, 193, 59–72.

CHEN, F. et SONG, K. (2009). Minimizing makespan in two-stage hybrid cross docking scheduling problem. *Computers & Operations Research*, 36, 2066–2073.

CHEN, P., GUO, Y., LIM, A. et RODRIGUES, B. (2006). Multiple crossdocks with inventory and time windows. *Computers & Operations Research*, 33, 43–63.

CHMIELEWSKI, A., NAUJOKS, B., JANAS, M. et CLAUSEN, U. (2009). Optimizing the door assignment in ltl-terminals. *Transportation science*, 43, 198–210.

COHEN, Y. et KEREN, B. (2009). Trailer to door assignment in a synchronous cross-dock operation. *International Journal of Logistics Systems and Management*, 5, 574–590.

DONALDSON, H., JOHNSON, E. L., RATLIFF, H. D. et ZHANG, M. (1998). Schedule-driven cross-docking networks. *Georgia tech tli report, The Logistics Institute, Georgia Tech.*

GUE, K. (1999). The effects of trailer scheduling on the layout of freight terminals. *Transportation Science*, 33, 419–428.

HANSEN, P. et MLADENOVIĆ, N. (2003). Variable neighborhood search. *Handbook of metaheuristics*, Springer. 145–184.

KINNEAR, E. (1997). Is there any magic in cross-docking ? *Supply Chain Management : An International Journal*, 2, 49–52.

KONUR, D. et GOLIAS, M. M. (2013). Cost-stable truck scheduling at a cross-dock facility with unknown truck arrivals : A meta-heuristic approach. *Transportation Research Part E : Logistics and Transportation Review*, 49, 71–91.

LARBI, R., ALPAN, G., BAPTISTE, P. et PENZ, B. (2011). Scheduling cross docking operations under full, partial and no information on inbound arrivals. *Computers & Operations Research*, 38, 889–900.

LEE, Y. H., JUNG, J. W. et LEE, K. M. (2006). Vehicle routing scheduling for cross-docking in the supply chain. *Computers & Industrial Engineering*, 51, 247–256.

LI, Y., LIM, A. et RODRIGUES, B. (2004). Crossdocking—jit scheduling with time windows. *Journal of the Operational Research Society*, 55, 1342–1351.

LIAO, T., EGBELU, P. et CHANG, P.-C. (2012). Simultaneous dock assignment and sequencing of inbound trucks under a fixed outbound truck schedule in multi-door cross docking operations. *International Journal of Production Economics.*

LIM, A., MA, H. et MIAO, Z. (2006). Truck dock assignment problem with time windows and capacity constraint in transshipment network through crossdocks. *Computational Science and Its Applications-ICCSA 2006*, 688–697.

MAKNOON, M. Y. (2007). *Simultaneous sequencing of incoming and outgoing semi-trailers on a cross-docking platform.* Mémoire de maîtrise, École Polytechnique de Montréal.

MAKNOON, M. Y. et BAPTISTE, P. (2009). Cross-docking : increasing platform efficiency by sequencing incoming and outgoing semi-trailers. *International Journal of Logistics : Research and Applications*, 12, 249–261.

MCWILLIAMS, D. (2009). A dynamic load-balancing scheme for the parcel hub-scheduling problem. *Computers & Industrial Engineering*, 57, 958–962.

MCWILLIAMS, D., STANFIELD, P. et GEIGER, C. (2005). The parcel hub scheduling problem : A simulation-based solution approach. *Computers & Industrial Engineering*, 49, 393–412.

MCWILLIAMS, D., STANFIELD, P. et GEIGER, C. (2008). Minimizing the completion time of the transfer operations in a central parcel consolidation terminal with unequal-batch-size inbound trailers. *Computers & Industrial Engineering*, 54, 709–720.

MIAO, Z., LIM, A. et MA, H. (2009). Truck dock assignment problem with operational time constraint within crossdocks. *European journal of operational research*, 192, 105–115.

OH, Y., HWANG, H., CHA, C. et LEE, S. (2006). A dock-door assignment problem for the korean mail distribution center. *Computers & Industrial Engineering*, 51, 288–296.

PECK, K. E. (1983). Operational analysis of freight terminals handling less than container load shipments. Rapport technique.

RATLIFF, H. D., VATE, J. V. et ZHANG, M. (2003). Network design for load-driven cross-docking systems [eb/ol]. Rapport technique, GIT Technical Report, the Logistics Institute, Georgia Institute of Technology, Atlanta, GA.

SADYKOV, R. (2012). Scheduling incoming and outgoing trucks at cross docking terminals to minimize the storage cost. *Annals of Operations Research*, 201, 423–440.

SCHWIND, G. F. (1995). Considerations for cross docking. *Material Handling Engineering*, 50, 47–49.

SOLTANI, R. et SADJADI, S. (2010). Scheduling trucks in cross-docking systems : A robust meta-heuristics approach. *Transportation Research Part E : Logistics and Transportation Review*, 46, 650–666.

STEPHAN, K. et BOYSEN, N. (2011). Cross-docking. *Journal of Management Control*, 22, 129–137.

TSUI, L. et CHANG, C. (1992). An optimal solution to a dock door assignment problem. *Computers & Industrial Engineering*, 23, 283–286.

TSUI, L. Y. et CHANG, C.-H. (1990). A microcomputer based decision support tool for assigning dock doors in freight yards. *Computers & Industrial Engineering*, 19, 309–312.

VAHDANI, B. et ZANDIEH, M. (2010). Scheduling trucks in cross-docking systems : Robust meta-heuristics. *Computers & Industrial Engineering*, 58, 12–24.

VAN BELLE, J., VALCKENAERS, P. et CATTRYSSE, D. (2012). Cross-docking : State of the art. *Omega*.

WALLER, M. A., CASSADY, C. R. et OZMENT, J. (2006). Impact of cross-docking on inventory in a decentralized retail supply chain. *Transportation Research Part E : Logistics and Transportation Review*, 42, 359–382.

WANG, J.-F. et REGAN, A. (2008). Real-time trailer scheduling for crossdock operations. *Transportation Journal*, 5–20.

YAN, H. et TANG, S.-L. (2009). Pre-distribution and post-distribution cross-docking operations. *Transportation Research Part E : Logistics and Transportation Review*, 45, 843–859.

YU, W. et EGBELU, P. (2008). Scheduling of inbound and outbound trucks in cross docking systems with temporary storage. *European Journal of Operational Research*, 184, 377–396.

ZHANG, M. (1997). Cross-docking network design.

ZINN, W. (1994). Cross docking : Hyperactivity on the dock. *Material Handling Engineering. p*, 57–58.