2011

Data mining framework for batching orders in real-time warehouse operations

Humberto Fuentes Saenz
Iowa State University

Data mining framework for batching orders in real-time warehouse operations

by

Humberto Fuentes Saenz

A thesis submitted to the graduate faculty

in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Major: Industrial Engineering

Program of Study Committee:

Sigurdur Olafsson, Major Professor

Yoshinori Suzuki

Jo Min

Iowa State University

Ames, Iowa

2011

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

Warehouse activities play a key role in the final customer service level. From the warehouse processes, order picking is the major contributor to this category overall expenses. Order batching is commonly employed to improve the resources efficiency. Several heuristics have been proposed for the order batching problem, most of them developed for static batching, although scarce research has been focused on dynamic batching via stochastic modeling.

We present an a novel approach to the problem developing a framework based on machine learning application directly to historical order batches data; gaining valuable knowledge regarding how are the batches formed and what attributes are the most meaningful in this process. This knowledge is then translated into simple batching decision rules capable of batch orders in a real-time scenario (dynamically). The framework was compared to FCFS heuristics and single picking; the results indicate higher performance.

CHAPTER 1. INTRODUCTION

The distribution of finished goods to end customers is playing an arduous role within the current global market. The present trend of business is reduce the delivery time to the lowest possible, combined with a high array of options available (i.e. mass customization), and smaller and more frequent orders. As a result of this trend, the necessity for fast and reliable warehouses and distribution centers is essential for the organizations aspiring to succeed among competitors.

A warehouse should be considered as the direct contact with the final customer by the organization since this is the final stage of their supply chain, in other words, the overall level of service of the company as perceived by the customer. The customer appreciation for the product provider will remain in three basic observations: the receiving of the right product, on-time delivery, and that the item fulfills all the expectations in functionality. These observations are mostly focused both in the manufacturing area as well as the supply chain area of the organization which are jointly required to excel in their processes in order to achieve a competitive advantage for the organization.

In the past, manufacturing companies had the common practice of assign a vast amount of resources for manufacturing processes development and improvement, perhaps diminishing potential benefits and savings in the supply chain area. Nowadays however, it can be observed that companies are directing more resources to the supply chain area. Therefore, it was not surprising that warehousing systems obtained a substantial interest

in the literature. The appeal to conduct research related to warehouse operations started in 1970 (Van Den Berg, 1999).

In theory, a warehouse should not be necessary according to principles dictated by Lean Manufacturing and Just In time philosophies, in real life however; unavoidable situations are presented in a manufacturing company creating a gap that only a buffer between finished goods and the final customer can solve. Some of these situations are: "to accommodate variability caused by seasonality, batching in production or transportation or value added processing such as kitting, labeling and product customization" (Gu et al., 2007). Despite the reason of why the warehouse is required in a company, it has been observed that five basic functions are performed in every warehouse, namely: receiving, put-away, storage, order picking, sorting, and shipping.

The literature agrees that the function of order picking has the highest potential for improvements from the warehouse basic operations. As mentioned by Ackerman (1997), it is the largest single expense category in the operation; hence the appealing for improvement. The order picking process consists of an order picker or an automated storage and retrieval machine taking a customer order and traveling from the depot to each of the order-lines (items) location in the warehouse, depositing in the retrieval cart or machine and transferring the items to the depot/shipping area. Tompkins et al., (2003) recognized three methods for picking orders as the most important: Single order picking, batch picking and zone picking. The present study is focused on the batching orders method which is based on the premise of combining a certain number of orders in one

single trip through the warehouse intending to save distance picking the orders together instead of individually.

1.1 Problem Statement.

The problem of batching orders in a warehouse environment is an extremely difficult task due to its nature. This complexity is detected when measuring the benefit or gain of merge an order to an existent batch since this measure is conditioned to the other orders in the batch (Rosenwein, 1996). Consequently, there is no uncomplicated mathematical model that can capture entirely the nature of this problem.

Literature on the order batching problem provides an array of algorithms to solve the problem of batching orders in a warehouse or distribution center (see Gibson and Sharp, 1992; Rosenwein, 1996; De Koster et al., 1999; Gu et al, 2007). However, a shortcoming perceived from these studies is the requirement of having all of the orders present at the time of initiate the algorithm. Moreover, some authors present procedures that can be processed several times during the day, although these procedures are computational-time consuming; hence, leading to a gradual loss of interest by the high end user as a result of the complexity of these algorithms.

Warehouse administrators commonly manage their picking operations either according to simple heuristics (De Koster 1999) i.e. first come first serve (FCFS) where first *n* orders are batched until the capacity of the storage and retrieval (S/R) facility is reached; or simply neglect the potential benefits of batch orders and perform single order

picking as a regular routine. As a result of this, we identified the problem as: develop a decision making tool that permits the processing of order batches in a real time basis, yet user friendly enough to stand as an operative tool for the warehouse administrator.

1.2 Research Objective.

The fundamental target to achieve is to understand the principles of why, when and how an order is batched to another order and translate this to an understandable and easy to interpret method, fast enough to provide a solution in a real time warehouse scenario and prone to be replicated to any general set of orders. In order to achieve this objective, we attempt to develop a framework derived with the application of data mining techniques to information obtained from simulated data batches, grouped with an existent batching heuristic.

1.3 Thesis Organization.

The present research is structured by five chapters. The subsequent four chapters are organized according to the following criteria: In chapter two the previous literature addressing the topic will be discussed. In chapter three the framework of concepts and techniques applied to obtain a solution for the problem will be presented. In chapter four the procedure applied in numerical examples with simulated data in three different warehouse scenarios will be presented along with performance results and a comparison of the procedure among commonly used heuristics. Finally, chapter five will present the conclusions regarding this study, its limitations and future course of research.

CHAPTER 2. LITERATURE REVIEW.

2.1 Warehousing.

Warehouse is commonly defined as the physical space where the storage of raw material and/or finished goods occurs. Some purposes for a warehouse are, to balance fluctuations between production and sales creating a buffer between both operations, consolidate the organization shipping activities in an effort to reduce logistics expenses or even as a tactic to provide faster customer deliveries (Tompkins et al., 2003).

A warehousing system is characterized as a collection of stochastic activities by its own nature, e.g. a common warehouse keep items coming from diverse suppliers with different estimated time arrivals (ETA) not always accurate, the receiving of raw material is dependent not only on the suppliers distance but also on the transportation method selected, the uncertainty of receiving customer orders as well as which items will be included on it; these are examples of common circumstances existent in every warehouse. Additionally, fast responsiveness and rapid adaptation to changes are required in a warehouse operation in order to sustain a strategic supply chain exceling in precise delivery of goods to the customer. Consequential to the settings stated before, activities performed by human operators prevail rather than high automated systems.

There are four activities performed in a warehouse that are considered as the most important: receiving, storage, picking and shipping. Moreover, picking activity is ranked as the highest expensive activity in a warehouse, as reported by Rosenwein (1996), Ruben and Jacobs (1999), Van den Berg (1999). Authors set these expenses as 55 percent

of operational costs (Drury 1988), and as high as 65 percent as stated by Coyle et al., (1996). The elevated expenses in the picking operations are a consequence of the significant amount of human labor required.

2.2 Order picking and batching methods.

The order picking process is defined as the collection of items from designated storage locations, furthermore, is the activity in which the warehouse is planned upon (Tompkins et al. 2003). Given the fact that the highest operational costs in the warehouse are allocated in this operation, hence it is expected that a reasonable amount of research in this topic would be available in the literature (see Ratliff and Rosenthal, 1983; Cormier and Gunn, 1992; De Koster et al., 2007).

Different approaches to fulfill the order picking process have been reviewed in the literature, according to Tompkins et al., (2003) the most representative are:

- Single order picking. The picker completes every order individually and sequentially.
- Batch picking. The picker completes several orders simultaneously.
- Zone picking. The warehouse is divided in zones, where one picker is assigned and collect the items located in the mentioned zone, either by individual orders or in batches.

From these options, batch picking is often preferred due to its broader applicability in several warehouse settings (see Tsai et al., 2008; Ruben and Jacobs, 1999). The objective

of batching orders pursues the minimization of picking efforts (regularly expressed in either distance or time metrics) in conjunction with a maximization of the S/R machine efficiency. As mentioned by Gibson and Sharp (1992), the rationale of this method is that combining two orders to be picked together, lead to reduced travel distances than single picking both orders.

The batching problem is considered exceptionally complex to solve and the computational time becomes a problem quickly for large number of orders; as indicated by several authors (see Elsayed and Unal, 1989; Chen and Wu, 2005; Rosenwein, 1996). Moreover, it is difficult to formulate with a mathematical program since the gain obtained by batching an order to a picking tour is conditioned to the orders that are already assigned to the tour (Rosenwein, 1996). Therefore, optimization approaches for the order batching problem are scarce, in contrast with the research focused on heuristic methods (Chen and Wu, 2005).

A classification for heuristic methods is proposed by De Koster (1999);

- Simple straightforward methods. Such as first-come first-served (FCFS) where *n* orders are batched sequentially upon arrival and until the S/R machine capacity is reached, then a new batch is started. Commonly used for comparison purposes.
- Seed algorithms. An order is selected as the batch seed, then, orders are added one by one until capacity is reached. Methods for seed selection and adding of orders are varied.
- Saving algorithms. Based on the time saved by batching two orders, compared to their single picking time.

Some of these heuristics are discussed in the following paragraphs.

Elsayed and Unal 1989 developed four order batching algorithms targeted to maximize distance savings in automated storage and retrieval systems. EQUAL algorithm starts by analyzing every possible pair of orders and choosing the pair that provides the largest distance savings as the seed, from then it will analyze the rest of the orders adding the order that provides greater time savings until the vehicle capacity is reached. The other tree heuristics, namely small-large SL, MAXSAV and CWright algorithms, follow the same principle of selecting an order pair as the seed, then, adding the order that maximizes savings in distance until capacity constraint is reached, with minor variations such as incorporate the volume of the order as a metric, analyze every possible combination of orders, and the application of a vehicle routing algorithm respectively. The SL algorithm provided the best distance savings.

Gibson and Sharp 1992 approached the order batching problem developing heuristics that group orders according to their proximity, the measurements accounted are the 4-dimensional spacefilling curve (SFC), and sequential minimal distance (SMD). They created 24 datasets for each heuristic, totaling 72 combinations, the order varied from 100 to 1200 and the results are expressed in terms of four different distance metrics: Euclidian, rectilinear, chebyshev and aisle travel. Their results indicate savings up to 44% compared to FCFS when SFC or SMD heuristics are used combined with an ABC storage policy.

Pan and Liu (1995) analyzed four order seed selection methods in an AS/RS warehouse system, this methods included; largest item number, largest weight of items, largest economic convex hull area, and smallest deviation from zero using a 6-D SFC

algorithm. They combined this order selection rules with four techniques for adding orders to the batch seed: largest number of common locations with the seed, minimum total distance between items closest locations, largest similarity coefficient of economic convex hulls values, and smallest deviation from the 6-D SFC Θ value. This combination results in 16 algorithms, additionally, the small-large heuristic discussed in Elsayed (1989) is also included. They formed batches using these 17 algorithms in simulated data experiments varying WH shape factor, S/R machine capacity and storage assignment type. Results indicate that the largest economic convex hull area method provides the best performance.

Rosenwein (1996) investigated two algorithms using also the principle of group orders according to their proximity, two measure distances are proposed, namely minimum additional aisle (MAA), and center of gravity (COG). They compare the results with the Gibson and Sharp heuristic and found that the best performance is obtained using the MAA distance.

Ruben and Jacobs 1999 researched batching heuristic applicable to multiple aisle warehouses, their heuristics objective is to assign zone number to orders according to the aisles required to pick the entire order e.g. the lowest number is assigned to an order requiring to visit only the first aisle whereas in a highest zone number the entire warehouse aisles are to be visited. The leading heuristic was the first-fit envelope based batching (FF-EBB), ranked in the greatest savings among several warehouse scenarios.

A few studies follow different methods, namely genetic algorithms, data mining techniques, e.g. Tsai et al 2008 propose a batching heuristic combining two genetic algorithms namely GA_Batch which produce batches minimizing travel cost and

earliness-tardiness penalties and the GA_TSP that targets to find the optimal travel path that minimizes the distance. The results are compared to another two algorithms targeting the travel cost minimization and the earliness and tardiness penalties cost respectively. The superior results were obtained with an algorithm targeting both targets simultaneously. Chen and Wu (2005) developed an approach based on mining customer demand patterns to discover frequent itemsets, followed by a 0-1 integer programming. Their results performed better than FCFS heuristic.

From previous research, it is perceived that some disconnection between academic research and warehouse real-life environments is experienced: despite the potential benefits that can be achieved by means of batching orders using the state-of-the-art techniques in warehouses, the administrators are inclined to employ simple methods such as FCFS for their picking operations (De Koster et al, 1999; Dekker et al., 2004). Moreover, Gu et al., (2010) affirmed that further research leading to expedite decisions and simple application is essential for the order batching problem. Our study is motivated by practical application; therefore, it is focused on these targets.

CHAPTER 3. METHODOLOGY

3.1 Introduction.

As revised in the previous literature, the problem of batching orders in a warehouse often leads to especially complex algorithms and procedures barely used by warehouse administrators due to its time consuming characteristics and assumptions inconsistent with a real-life warehouse environment, e.g. consider that all of the orders are known in advance to produce the batches for the period. This is an assumption hardly observed in a real-time warehouse environment, where immediate decisions are required in accordance to their operational problems. Hence, the attempt of this study is to provide a framework to obtain a decision tool to form order batches that is easy to interpret, providing a practical decision-making instrument for the warehouse administrator.

The present study is based in the framework developed by Li and Olafsson (2005), where the authors applied data mining techniques to several job schedules in order to obtain both dispatching rules as well as structural insights previously unknown (implicit and explicit knowledge). We modify this framework through the application of data mining techniques to existent order batches data, previously grouped with an order batching heuristic. Through the framework implementation, we identified potential benefits:

- Process the information gained during a previous time period (static batching), transforming it into decision-making tools applicable for subsequent real-time periods (dynamic batching).
- Improve the performance attained with naive methods (FCFS, single picking).

- Provide fast responsiveness and usage simplicity; characteristics consistent with a warehouse environment, thus, expected to be employed regularly.

3.2 Real-time batching decision-making tool framework.

In this section, we present the methodology for knowledge discovery from previous order batches information. We based our framework on one of the most common application of data mining: classification, where the goal is to discriminate examples of data (e.g. instances) into a class value (Olafsson et al., 2008), this discrimination is based on the attributes values observed in each instance. Thus, we identify our class value (i.e. target concept to be learned) as; upon receiving of two orders, decide whether to batch the orders or not. This knowledge would permit grouping orders into batches in a real-time warehouse environment by comparing two orders focusing only on meaningful attributes, therefore, improving performances achieved through heuristics like FCFS.

In order to accomplish the target concept, we can divide our framework in three main phases, as follows:

1. Obtain information from previous batches formed. First, simulated data is generated for three different warehouse scenarios; afterwards, the FF-EBB heuristic is implemented, forming the batches of orders.
2. Data preprocessing for the learning algorithm induction. Procedures included in this phase include data modifications necessary for the flat file construction, (i.e. create pairwise order comparisons as learning instances, assign class values, attribute selection, and attribute construction).

3. Learning algorithm induction and transformation into graphic tool to assist in order batching decisions. These phases are portrayed in Figure 1.

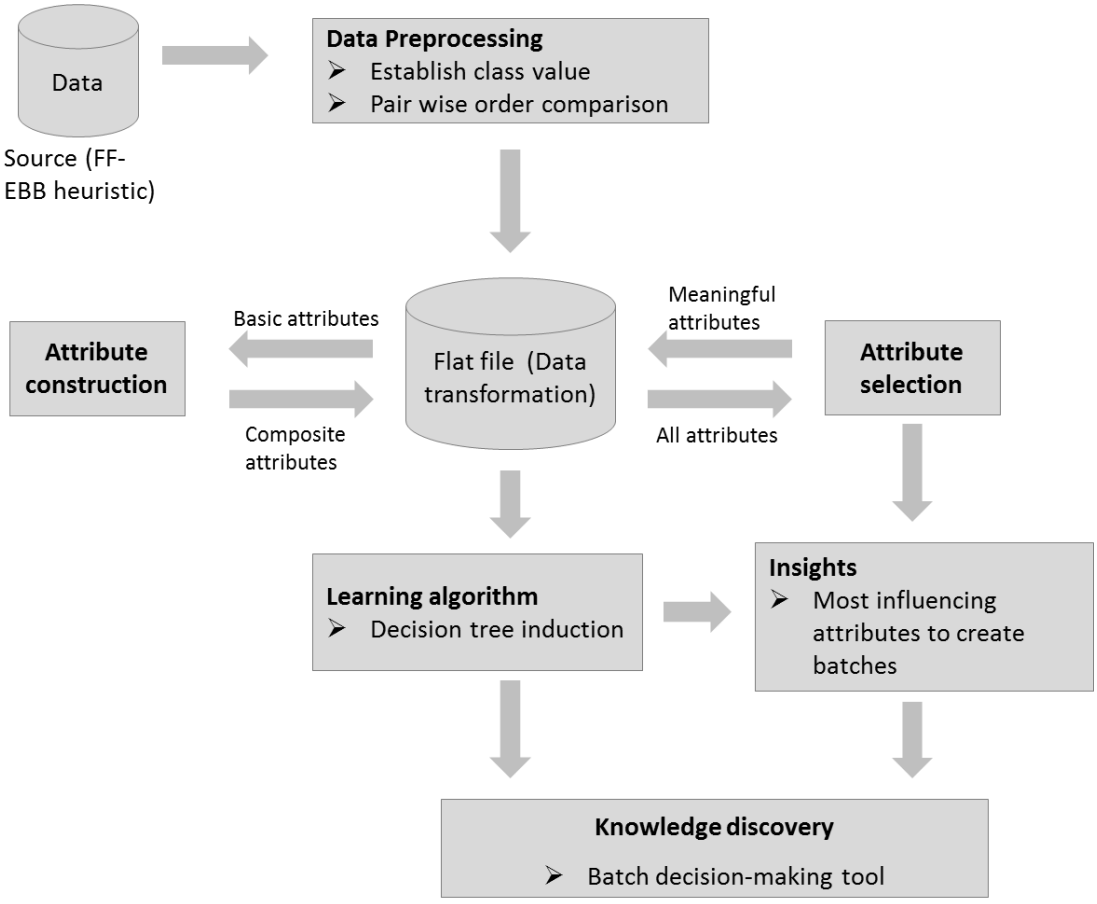

Figure 1. Adapted framework for order batching knowledge discovery.

Consistent to the framework described before, the first phase is crucial for valuable knowledge acquirement, in fact, the results eventually found would be a reflection of this source; in other words, this is the underlying structure supporting the decisions and insights provided by our framework. Given that our study is focused in developing a

procedure for knowledge discovery and a decision making tool rather than create a new method to create order batches directly, we select a previous heuristic to form order batches from simulated data that will supply information for our flat file. The heuristic mentioned had to accomplish some prerequisites considered necessary for the results searched:

- Applicable in a wide-range of warehouse settings.
- Outperforming heuristics in the same category (general application).

Such heuristic was identified in the first-fit envelope based batch (FF-EBB) developed by Ruben and Jacobs (1999). According to Bozer and Kyle, (2008) this heuristic generally meets the two conditions.

To this point we have two different sources of data: the orders database along with the batches formed from this database. Thus, according to the knowledge discovery (data mining) process, the next step is data preprocessing. Furthermore, as indicated in Olafsson et al. (2008), we have to execute a significant amount of preprocessing prior to a learning algorithm accurate induction. Then, it is necessary to integrate these two sources of data into a single flat file. Besides this integration, we have to model the flat file according to basic classification process requirements: The file has to be built by a series of examples (from now on referred as instances), each of this instances is described by attributes and belong to a predetermined target class. This is the foundation of supervised learning for classification (Han and Kamber, 2006).

After modeling the data in a flat file proper for a learning algorithm induction, it is required to identify what are the meaningful attributes to include in our database for knowledge extraction. We found very improbable that the inherent attributes characterizing an order (e.g. order ID, arrival time, items requested and quantities), will offer us meaningful knowledge or insights that improve the batching decision process. Hence, the creation of new attributes is necessary, mainly, these can be created intuitively by exploring the interactions of the order with the warehouse settings, a few examples are; aisles traveled in the tour, total picking distance. We also noted that a process that would be beneficial for our framework, namely attribute selection. Moreover, insightful knowledge can be obtained by selecting what attributes are the most important in our model (Li and Olafsson, 2005).

Ultimately, our data is ready to be mined by a learning algorithm. There is a broad range of methods used for classification, e.g. decision trees, neural networks, support vector machines, and bayesian networks (Olafsson et al., 2008). Specifically to our framework, the objective is identified as a decision-making tool designed to be used in a fast environment; therefore, such tool was chosen as a decision tree algorithm. Main characteristics of decision trees are; easy to comprehend, provide inherent insights about meaningful attributes (attributes used to construct the tree), and the availability of implementations. So, even after considering that usually does not provide the best accuracy, its connection with our objective entices its usage.

The resulting decision tree can be used as a graphical tool to assign order into batches in a real-time basis. The target concept was identified as; comparing two orders, decide promptly whether to batch the orders or not. Therefore, orders arriving at the warehouse in a real-time basis can be batched following decision rules easily extracted from the tree.

3.3 Numerical Example.

Once the framework was defined in the previous section, a numerical example is considered to be significantly helpful for its proper understanding. This example will be carried out correspondingly to the framework stages and is presented in the following five sections.

3.3.1 Orders data generation.

Our framework is initialized with a database of orders; the simulated data is generated using parameters that in some way reflect those expected in a warehouse real environment. Also, the following assumptions were considered for data generation:

- Number of orders is fixed and equal to 10 orders.
- Number of items contained in each order is uniformly distributed over [1, 4].
- Quantity requested for each item is assumed to be one.
- Number of different items administered in the warehouse is 240.
- Items are randomly assigned in the warehouse, so that each item is equally likely to be requested.

- The orders arrive uniformly distributed in a one hour time period.

The following dataset is generated using the assumptions previously specified. The real-time framework will be performed in the dataset shown in Table 1.

Table 1. Orders dataset for numerical example.

Arrival time	Order ID	Items requested				Volumen
8:08	O1	156				1
8:14	O2	89				1
8:16	O3	15	57	139		3
8:18	O4	11	41			2
8:21	O5	24	32			2
8:23	O6	12	101	217	227	4
8:26	O7	165				1
8:35	O8	179				1
8:52	O9	132				1
8:59	O10	27	89	188		3

The table describes the basic information of orders arriving to the warehouse; the implicit information contained in each order will be used to batch the orders in an efficiently manner.

3.3.2 Warehouse layout and assumptions.

In order to create batches of orders from the database generated in the previous section, we first have to match the orders information with a specific warehouse layout. The most visited warehouse scenario in the literature is represented by the single block layout (see Bozer and Kyle, 2008; Gademann et al., 2005; Rosenwein, 1996; Gibson and Sharp, 1992), described by multiple racks or material location bays arranged in parallel,

consequently forming parallel-aisles to pick the items. Two cross aisles are assigned to change from one aisle to another and these are located at the beginning and ending of the racks. After thorough consideration of alternatives, we distinguished a specific characterization of the single-block warehouse layout and storage-retrieval equipment reflecting those observed commonly in existing warehouses, presented in De Koster et al., (1999). Specifically, the warehouse settings assumed for the numerical example are presented in Table 2:

Table 2. Warehouse parameters for small example.

Parameter	Small example WH
Order size	U [1-4]
S/R machine capacity	7
Aisles number	4
Item locations per aisle	60
Total locations number	240
Aisle length	50 m
Trasversal distance between aisles	4.3 m

(Adapted from De Koster et al., 1999).

The specific single-block warehouse layout is presented in Figure 2 for improved visualization. Notice that the first three orders item locations from the database are displayed in the warehouse layout:

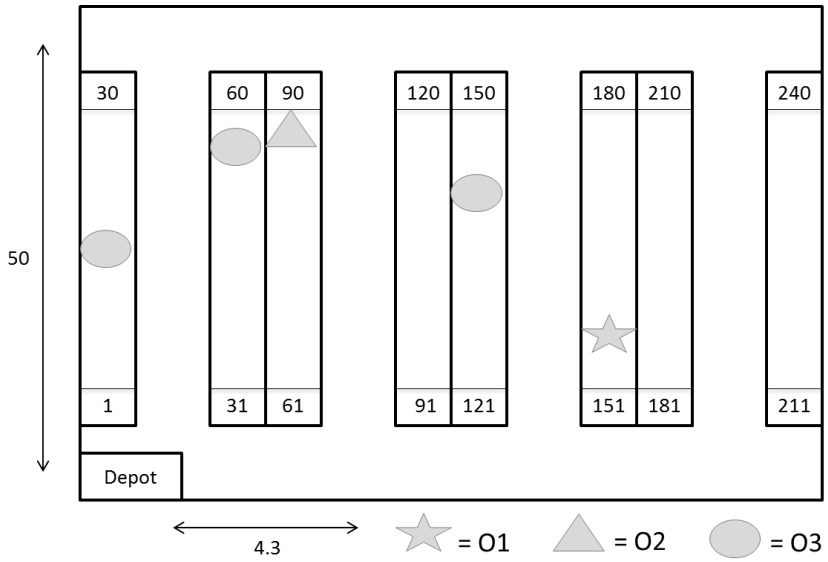

Figure 2. WH layout configuration for small example.

After the layout has been configured, operative assumptions describing the order picking process have to be acknowledged in advance of the order batching procedure application. This study is mainly motivated by practical application in different warehouse settings; hence, assumptions considered are the most associated with real-world warehouse scenarios:

- Performance is measured in terms of distance traveled.
- An order cannot be split (due to further sorting required and possible inaccuracies at shipping stage).
- An order is assumed to include fewer items than the S/R machine capacity.
- The total number of orders assigned in a batch is constrained by the capacity of the S/R machine which is seven items.
- Each item is assigned to a specific location in the warehouse.
- The S/R machine is able to travel the aisle in either direction.

- The picker follows an S-shape route (Hall, 1993) through the warehouse, entering an aisle whenever an item is located in it and traveling the entire aisle. An exception is considered when the number of aisles to visit is odd (the picker can return in that particular case).
- The depot is positioned at the beginning of the leftmost aisle in the warehouse.
- A batch retrieval tour starts in the depot, travels the aisles where items are located and ends in the depot releasing the items.
- Picking from both right and left aisles is considered.
- Only horizontal movement is considered in this study.
- Material stockouts are not considered.

3.3.2 FF-EBB heuristic as underlying structure.

The purpose of this study make an emphases in finding meaningful knowledge from order batches previously formed, hence, in order for this knowledge to be meaningful we need to find good solid batches leading to proven benefits in the picking operation performance in warehouses. We selected an existent heuristic for the order batching problem namely FF-EBB developed by Ruben and Jacobs (1999) since our study is focused in the framework to mine knowledge from batches information, rather than generate a heuristic model on its own. This heuristic was employed in simulated data to form order batches with the purpose of create a flat file data to mine afterwards.

The problem is described as, given a set of orders in the warehouse context previously described; how to group the orders in batches in order to reduce substantially the picking efforts required to retrieve all of the orders. The previous mentioned FF-EBB heuristic approaches this problem by sorting the orders in terms of a function involving the first and last aisle required to pick the entire order. All of the different combinations of first-last aisles (called min-max aisles respectively from now on) are contemplated, numbered and referred as "order envelope". Mathematical notation utilized in the FF-EBB heuristic is presented below (from Ruben and Jacobs, 1999),

M = Number of aisles in the warehouse

E = $\left(\frac{M}{2}\right) + M = \frac{1}{2}(M^2 + M)$

For our four aisle warehouse, there are a total of 10 envelopes. The method applied to number the envelopes follows an objective defined in the following formulations,

C_k = Cross distance necessary to retrieve all orders from order envelope k (k = 1,2,…,E)

Since the distance between aisles is the same, it is assumed that $C_k = (max_k - min_k)$ where min_k and max_k are the minimum and maximum aisle describing order envelope k.

$C_{kk'}$ = cross aisle distance resulting from batching two orders with different order envelope index = max $\{max_k, max_{k'}\}$ – min $\{min_k, min_{k'}\}$

$\delta_{kk'}$ = Symmetrical measure of increase in cross distance = max$\{(C_{kk'} - C_k),(C_{kk'} - C_{k'})\}$

The scheme used to number the envelope sets ensures that $\delta_{kk'} = 1$ for each pair of consecutively numbered envelope sets.

The envelopes are numbered as follows:

$$k = \begin{cases} \frac{max^2 + max}{2} - \min + 1 & if \max\ is\ even \\ \frac{max^2 + max}{2} - \max + min & if \max\ is\ odd \end{cases}$$

Thus, all order envelopes with max = 1 are numbered with a low index and considered first in the sorted list for inclusion into batches, followed by envelopes with max = 2 and so on. For a better appreciation of the envelope index numbering scheme Table 3 shows the sorted envelopes index along with the min – max aisles combinations.

Table 3. Order envelope scheme.

min aisle	max aisle	C_k	k
1	1	0	1
2	2	0	2
1	2	1	3
1	3	2	4
2	3	1	5
3	3	0	6
4	4	0	7
3	4	1	8
2	4	2	9
1	4	3	10

From Table 3, is easily identified that the FF-EBB heuristic aims to group orders according to the location of aisles required to pick the entire order, i.e. considering the

event that a set of orders is contained in the first three envelopes, the picking tour corresponding to this batch consists of visiting only the first 2 aisles. Therefore, the lowest envelope index denotes a first aisle trip whereas the highest envelope index might result in traveling all of the aisles in the warehouse.

The results (batches formed) of the FF-EBB heuristic applied in the example data are presented in Table 4 and performance comparisons among single picking, FCFS, and the referred FF-EBB heuristic is shown in table 5:

Table 4. Batches formed with the example database using FF-EBB heuristic.

Batch	Orders included	Volume	Picking distance	Aisles traveled	Cross aisles traveled	Facility utilization
1	O1, O2, O4, O5, O9	7	137.2	3	2	1
2	O3, O7, O8	5	117.2	2	2	0.71
3	O6, O10	7	182.4	3	3	1
		Total	436.8	8	7	0.9

Table 5. FF-EBB performance comparison VS single picking & FCFS.

Methods compared	Picking distance	Aisles traveled	Cross aisles traveled	Facility utilization
Single Picking/FF-EBB	0.50	0.53	0.41	0.30
FCFS/FF-EBB	0.74	0.73	0.88	1.00

Table 5 indicates the dominance of the FF-EBB heuristic employed to form order batches, compared to simple, naïve heuristics such as single picking or FCFS, resulting in travel tours 50% and 26% shorter respectively. The batches of orders can now be

translated into a single flat file prone to be mined. This transformation is explained in the following section.

3.3.4 Flat file creation.

Considering the orders dataset presented in Table 1 along with the batches formed with the orders shown in Table 2, we have sufficient information to create a flat file to mine implicit knowledge from the batches. The target concept was previously stated as; comparing two orders, decide promptly whether to batch the orders or not. Assuming we have the information of batches formed with these orders but we do not know what technique was used to group the orders (needless to say, the batches were formed with the FF-EBB heuristic). In correspondence with our target concept, we need a classification system to easily decide whether to batch orders or not, following the same criteria used to create the original order batches.

Thus we employ a pairwise comparison of all the orders contained in the database to create examples of batching decisions from which we can learn. An assumption we are making is that two orders will be compared to take a decision of whether to batch this two orders or not and that the order database is arranged according to their arrival sequence. Therefore, it is inferred that the first order can eventually be compared to the rest of the orders, the second order is compared to the third and to the rest and so on; leading to an order comparison scheme where every pair of orders contained in the database will eventually be compared.

An essential attribute defined for this file is the target concept; an attribute titled "Batch" is then created for which two values can be assigned: yes or no. The rationale behind this proceeding is that, when all the specific features of two order features are compared, we want to decide whether those orders should be picked together (Batch = yes) or not (Batch = no). Moreover, it is preferably that the framework also provides insights about what are the most conclusive attributes for grouping order batches, besides of interesting combinations of attributes that facilitate the decision-making process. Thus far we identified the class attribute; the remainder attributes are recognized as the ones that best describe the order characteristics. Hence, the attributes employed for the flat file construction are selected following the criteria mentioned before and are presented in Table 6.

Table 6. Attributes initially employed for flat file creation.

Attribute	Description	Type
order pair	ID for order pair considered	Nominal
aisles 1	Number of aisles required to pick order 1	Numeric
vol 1	Order 1 volumen	Numeric
sp dist 1	Single pick distance to pick order 1	Numeric
aisles 2	Number of aisles required to pick order 2	Numeric
vol 2	Order 2 volumen	Numeric
sp dist 2	Single pick distance to pick order 2	Numeric
batch?	Batch decision (yes, no)	Nominal

It is clearly seen than most of the attributes presented in Table 6 can be easily obtained from the order inherent information, only the single pick distance attribute requires further calculation, however, a spreadsheet table can be easily created for this purpose. The resulting table is presented in Table 7.

Table 7. Flat file transformation from order batches information.

order pair	aisles 1	vol 1	sp dist 1	aisles 2	vol 2	sp dist 2	batch?
O1 - O2	1	1	37.18	1	1	105.17	yes
O1 - O3	1	1	37.18	2	3	117.2	no
O1 - O4	1	1	37.18	1	2	36.63	yes
O1 - O5	1	1	37.18	1	2	6.66	yes
O1 - O6	1	1	37.18	3	4	182.4	no
O1 - O7	1	1	37.18	1	1	67.15	no
O1 - O8	1	1	37.18	1	1	113.77	no
O1 - O9	1	1	37.18	1	1	57.16	yes
O1 - O10	1	1	37.18	3	3	152.44	no
O2 - O3	1	1	105.17	2	3	117.2	no
O2 - O4	1	1	105.17	1	2	36.63	yes
O2 - O5	1	1	105.17	1	2	6.66	yes
O2 - O6	1	1	105.17	3	4	182.4	no
O2 - O7	1	1	105.17	1	1	67.15	no
O2 - O8	1	1	105.17	1	1	113.77	no
O2 - O9	1	1	105.17	1	1	57.16	yes
O2 - O10	1	1	105.17	3	3	152.44	no
O3 - O4	2	3	117.2	1	2	36.63	no
O3 - O5	2	3	117.2	1	2	6.66	no
O3 - O6	2	3	117.2	3	4	182.4	no
O3 - O7	2	3	117.2	1	1	67.15	yes
O3 - O8	2	3	117.2	1	1	113.77	yes
O3 - O9	2	3	117.2	1	1	57.16	no
O3 - O10	2	3	117.2	3	3	152.44	no
O4 - O5	1	2	36.63	1	2	6.66	yes
O4 - O6	1	2	36.63	3	4	182.4	no
O4 - O7	1	2	36.63	1	1	67.15	no
O4 - O8	1	2	36.63	1	1	113.77	no
O4 - O9	1	2	36.63	1	1	57.16	yes
O4 - O10	1	2	36.63	3	3	152.44	no
O5 - O6	1	2	6.66	3	4	182.4	no
O5 - O7	1	2	6.66	1	1	67.15	no
O5 - O8	1	2	6.66	1	1	113.77	no
O5 - O9	1	2	6.66	1	1	57.16	yes
O5 - O10	1	2	6.66	3	3	152.44	no
O6 - O7	3	4	182.4	1	1	67.15	no
O6 - O8	3	4	182.4	1	1	113.77	no
O6 - O9	3	4	182.4	1	1	57.16	no
O6 - O10	3	4	182.4	3	3	152.44	yes
O7 - O8	1	1	67.15	1	1	113.77	yes
O7 - O9	1	1	67.15	1	1	57.16	no
O7 - O10	1	1	67.15	3	3	152.44	no
O8 - O9	1	1	113.77	1	1	57.16	no
O8 - O10	1	1	113.77	3	3	152.44	no
O9 - O10	1	1	57.16	3	3	152.44	no

Table 7 shows certain inefficiency in the data according to the data mining requirements for proper learning algorithms induction:

- Imbalanced data.
- Dependency among instances.

In the first issue, the dataset is imbalanced since there are only 14 instances corresponding to the positive class (yes) in contrast to 31 instances corresponding to the negative class. The outcome of any learning algorithm applied in this data would possibly ignore the positive instances and predict simply a negative class for any new instance, still reaching almost a 70% in accuracy rate. Moreover, this issue is not really obvious in this particular dataset because it is constructed with a small order quantity; however, if the number of orders is increased it will certainly be augmented in consequence. In our study, we want to test the applicability of this framework in several warehouse scenarios congruent with real-life environments so that experimentations are performed with a total number orders as high as 100 orders, resulting in 4950 total instances, thus the imbalanced data problem is need to be addressed effectively.

This kind of dataset condition is encountered more often in real-world fields, e.g. network intrusion detection, revealing fraudulent credit card transactions, oil spill detection in satellite radar images (Kotsiantis, et al., 2006). In order to address this problem several techniques have been proposed in the literature, in our study, we considered the following methods; undersampling of the negative instances,

oversampling of the positive instances, and finally cost-sensitive learning. The classification performances of these techniques were tested in the original data in order to define what method best suits our objectives. Results are presented in Table 8,

Table 8. Decision tree accuracy results classifying original dataset.

Method	TP	FP	TN	FN	Accuracy	precision	recall	f-measure	DT nodes
Undersampling	13	12	19	1	0.71	0.52	0.93	0.67	3
Oversampling	9	2	29	5	0.84	0.82	0.64	0.72	4
Cost sensitive	9	2	29	5	0.84	0.82	0.64	0.72	2

The common measure of accuracy is not sufficient for imbalanced datasets, so that measures of precision, recall and f-measure combining both previous measures are also pondered to select the best technique to classify our data. From these methods, the best performance for classifying the data shown was achieved with cost-sensitive learning, moreover, this technique provided the same accuracy altogether with oversampling but we also account for the number of nodes required to classify an instance, thus less nodes mean fewer comparison between orders, so from now on this method is only considered in our framework.

Additionally, there is a requirement for flat files that each instance is an example of knowledge independent from all the others instances; as noticed in the flat file, there is dependency among instances due to the fact that we are forming order pair comparisons. However, we identified comparable studies and real-world practical applications

proceeding satisfactorily against this assumption of independence (Li and Olafsson, 2005).

A critical question arises from the flat file generated with batches information: What are the most contributing attributes to classify an instance as positive, in other words, to batch the pair of orders? Therefore, we proceed to induce a decision tree learning algorithm to our data, having different objectives in mind:

- Examine the classification capabilities of the learning algorithm.
- Visualize the most important attributes utilized to classify new instances, examining each level of the tree, i.e. the attribute at the top provides the greatest information gain from all and so on. Moreover, attributes combinations can be discovered by following each branch of the tree.

We used the WEKA software version 3.7.3 (Hall et al, 2009) for all of our data mining applications. The model resulting from the application of the C4.5 decision tree algorithm (Quinlan, 1993) is presented in Figure 3. We omit the methods for handling the imbalanced data problem resulting in lowest performances and present only the decision tree using the cost sensitive learning method.

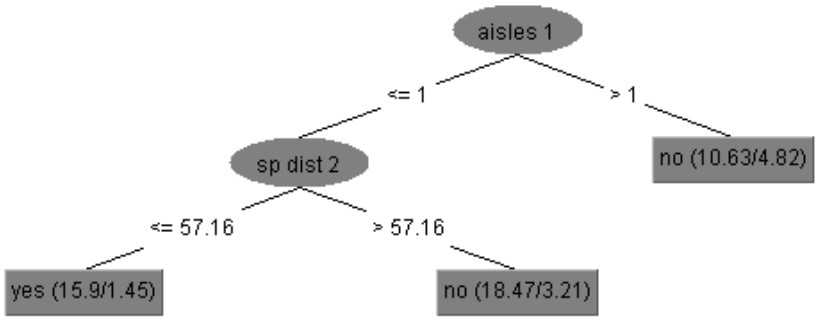

Figure 3. Decision tree outcome using cost sensitive learning.

The cost of misclassifying a positive instance was set as 2.21 whereas a negative instance misclassification is 1. (From the relation of negative to positive instances: 31/14 = 2.21).

There is a single decision rule for instance classification that can be drawn from the decision tree:

IF aisles 1 ≤ 1 AND sp dist 2 ≤ 57.16 THEN batch? = yes

Even though applying this single decision rule to the original dataset lead to an 84% in classification accuracy, the tree is not providing any insightful information about the order batching criteria used. Hence, we need to modify our flat file and include attributes considered to enhance the information provided by the tree, consequently, four attributes are added, defined as follows: *common aisles* (number of aisles traveled in both order 1 and order 2), *added aisles* (number of extra aisles to travel if order 2 is batched to order 1), *cross aisles* (total number of cross aisles traveled to pick the batch) and finally, *batch dist* (traveled distance to pick the batch). The C4.5 algorithm was then applied once more to the data including these four new attributes, the resulting decision tree is depicted in Figure 4.

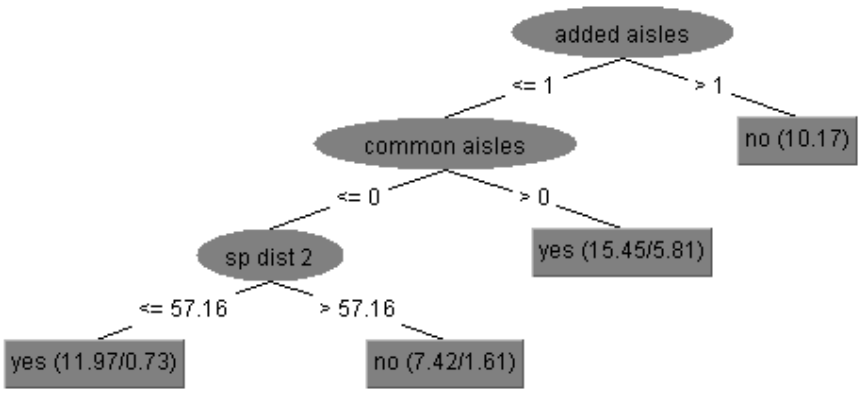

Figure 4. Decision tree outcome using four new attributes.

According to the target variable, a "yes" in a leave of the tree indicates that the pair of orders compared should be batched together. Hence, the decision tree can be used to batch any dataset of orders, also, this decision for batching orders can be grasped straightforwardly comparing either two or three order attributes (corresponding to the "yes" leaves in the tree).

Some other insights can be also drawn from it such as; what are the most important attributes considered to batch a pair of orders? The flat file included six attributes describing both orders, along with four attributes describing the batch of the orders; however, the decision trees make use of only three attributes to determine whether it is advantageous to batch the order pair or not, so that we can conclude this are the most meaningful attributes for this particular warehouse and these orders characterization.

The decision tree provides also some insights about the underlying structure used to batch the orders, e.g. the most important attribute was identified as "added aisles" making reference to the total number of aisles required to pick the batch if order 2 is

batched to order 1. In all, this logic is congruent with the FF-EBB heuristic being the underlying structure previously employed to batch the orders. As analyzed before, this heuristic employs a scheme of aisle numbering in an attempt to minimize the number of aisles required to pick the entire batch. Hence, this is confirmed by the first node in the decision tree: added aisles ≤ 1, denoting that the inclusion of the second order require either zero or one extra aisles to the picking route traveled to pick the first order. Needless to say, any kind of heuristic can be employed for a specific warehouse setting due to the framework flexibility, i.e. warehouses inclined more for expedite deliveries sacrificing efficiency on picking operations utilize a different criteria to form batches of orders, perhaps involving time periods; in that case, different results for most important attributes are expected along with decision trees more ad hoc to those requirements.

The following rules for a positive batch decision (batch? = yes), are extracted from the decision tree in Figure 4:

IF added aisles ≤ 1 AND common aisles > 0 THEN Batch = yes

IF added aisles ≤ 1 AND common aisles ≤ 0 AND sp dist 2 ≤ 57.16 THEN Batch = yes

Otherwise Batch = no

Considering these two rules, a decision to whether batch a pair of orders or not can be easily selected comparing attributes requiring minimal or no computing effort at all, e.g. added aisles and common aisles quantities can be identified directly reviewing the items included on the order pair, whereas single pick distance of orders could be

easily determined with a spreadsheet table; hence, the motivation to apply this framework to get straightforward, simple decision guidelines to batch orders could be appealing in the real-world warehouse scenario.

3.3.5 Decision tree as a batch decision-making tool.

To illustrate the framework performance, we present a new dataset of orders and proceed to form batches of orders based solely on the decision rules previously extracted from the trees and compare results afterwards versus the simple naïve heuristics FCFS and single picking. The orders in Table 9 are randomly generated following the same parameters as in the small example.

Table 9. New orders dataset.

Arrival time	Order ID	Items requested				Volumen	Aisles visited				Single pick dist
8:09	O1	99	143			2		2	3		117.2
8:12	O2	57	111	139		3	1	2	3		173.8
8:31	O3	41	55	75	105	4	1	2			108.6
8:33	O4	106	236			2		2		4	125.8
8:37	O5	50	73	141	150	4	1	2	3		217.1
8:46	O6	91	155	191	197	4		2	3	4	182.4
8:49	O7	8	63	193		3	1	2		4	169.1
8:51	O8	77	102			2		2			48.6
8:54	O9	54	171			2	1		3		117.2
8:56	O10	211				1				4	29.1

Notice the deliberate addition of the aisles to be visited in each of the orders as well as the single pick distance incurred if each order is picked on its own. The other metric number of common aisles in a pair of orders can be easily identified comparing

each of the aisles visited. Applying the two rules previously drawn from the tree, we proceed to group the orders in batches, contemplating also the S/R facility capacity constraint. The batches formed results are presented in Table 10.

Table 10. New batches formed with decision tree rules.

Batch	Orders included	Volume	Picking distance	Aisles traveled	Cross aisles traveled	Facility utilization
1	O1, O2, O4	7	225.8	4	3	1
2	O3, O7	7	169.13	3	3	1
3	O5, O8, O10	7	225.8	4	3	1
3	O6, O9	6	225.8	4	3	0.86
		Total	846.53	15	12	0.96

The total picking distance traveled is the most important metric given our objective; however, it is necessary to provide a point of comparison to discern the performance achieved. Thus, the following Table 11 presents a comparison in performance metrics resulting from the decision tree, FF-EBB heuristic and simple naïve heuristics application in the same data:

Table 11. Results comparison among order batching methods.

Methods compared	Picking distance	Aisles traveled	Cross aisles traveled	Facility utilization
FF-EBB / Decision tree	0.92	0.87	0.75	1
FCFS / Decision tree	1.22	1.07	1.08	0.8
Single picking / Decision tree	1.52	1.47	1.83	0.4

As presented in the table, there is a decrease of 8% in efficiency by forming the batches using the decision tree versus the FF-EBB heuristic. The increasing of distance traveled was somehow expected, given that the decision tree is basically emulating the heuristic methodology, so that an increasing in efficiency is not to be expected, on the contrary, the expected performance of the decision tree would be to match the traveled distance reached using the FF-EBB heuristic. On the other hand, the performance of the batching rules drawn from the decision tree versus FCFS and single picking heuristics, results in 22% and 52% shorter routes respectively. Also, the rest of the metrics such as aisles traveled, cross aisles traveled and facility utilization are improved compared to FCFS and single picking.

Some remarks can be derived from the analysis of the experiment results, i.e. historical batch information from a specific warehouse can be employed to batch new instances in a real-time basis, the decision tree learning algorithm applied to this information can extract the underlying structure used to create batches of orders, in other words, mine the criteria utilized to batch the orders and use this knowledge to batch new orders following simple rules. This mined knowledge can be predictive as well as descriptive. Predictive referring to the capacity of deciding if the orders should be batched and descriptive according to understand why the orders are batched and what are the most important attributes leading to this decision. The warehouse goal will indeed be reflected on the decision tree outcome, for instance, using the underlying structure heuristic selected in this study lead to rules focusing solely in distance reduction as a

goal, whereas a different warehouse can be driven by expedited shipments (shorter delivery times) or a combination of both schemes.

Extended numerical experimentation will be presented in the following chapter to further investigate the framework performance on diverse warehouse settings as well as different order quantities.

CHAPTER 4. NUMERICAL RESULTS.

In the previous chapter, we presented potential advantages of decision tree learning algorithm applied to data obtained from warehouse order-batches information. This information can be acquired using static-batching heuristics applied in historical data and then transformed into comprehensible decision rules to determine orders batches. The results drawn from the small example entices a further analysis of the framework performance in extended scenarios, therefore, we set up a series of experiments in the following section.

4.1 Framework simulation experiments set-up.

In this section, a series of experiments are presented to analyze the framework performance in different warehouse layout scenarios as well as different orders sizes. The warehouses considered in the study are identified as commonly encountered in real-life operative warehouses in terms of layout, dimensions, and order picking facility equipment (De Koster, 1999). The three warehouse layouts are depicted in Figure 3 for reference.

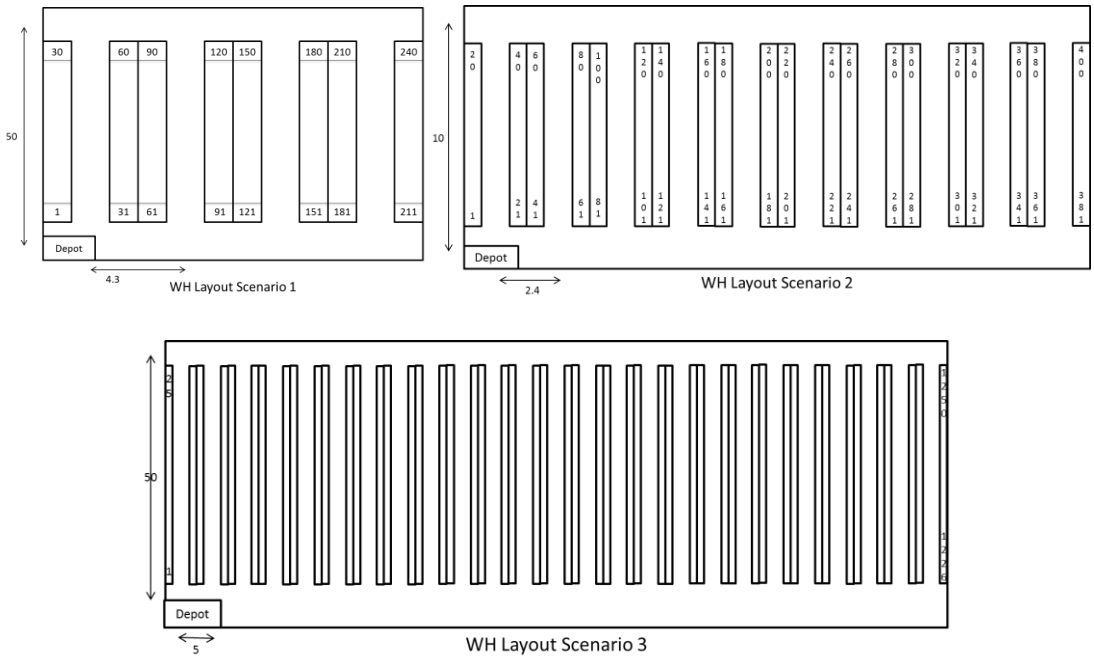

Figure 5. Warehouse layout scenarios employed in experiments.

The warehouse first scenario is commonly encountered in narrow-aisle pallet warehouses, the second scenario is similar to shelf stores with manual pickers and the third scenario is compared to a pallet warehouse in terms of size only. Each one of these layouts presents different characteristics in terms of measures, item quantities included in an order, aisles number, facility capacity and so forth. These characteristics are portrayed in Table 12.

Table 12. Warehouse layout scenarios characteristics.

Parameter	WH Scenario 1	WH Scenario 2	WH Scenario 3
Order size	U [1-4]	U [2-10]	U [5-25]
S/R machine capacity	7	24	150
Aisles number	4	10	25
Item locations per aisle	60 (2 x 30)	40 (2 x 20)	50 (2 x 25)
Total locations number	240	400	1250
Aisle length	50 m	10 m	50 m
Trasversal distance between aisles	4.3 m	2.4 m	5 m

As it is observed in the table, we include warehouses settings increasing progressively in aisles, item quantities and total item locations to examine performances among various capacities. The orders were generated using MATLAB software, using discrete uniform distributions with identical probabilities for both order sizes as well as the item identifier. We considered three different order quantities, ranging from O = 30 and up to O = 100 orders. A total number of four replications were considered for each combination of warehouse layout and order quantity. In overall we included a total of experiments = 3 warehouse scenarios * 3 order quantities * 4 replications = 36 experiments.

Foremost, our framework attempts to reproduce the batches of orders formed with the FF-EBB heuristic in a new dataset with the objective to reduce the overall traveled distance as much as possible; hence, our first measurement is to define how well performs the total distance traveled by batching the orders with the decision tree against the overall distance traveled picking the orders batched with the heuristic, the results are presented in Table 13.

Table 13. Accuracy average percentage to replicate FF-EBB heuristic travel distance with decision tree.

WH Scenario	Number of orders	Average accuracy (%)	Minimum accuracy (%)	Maximum accuracy (%)
1	30	94.56	90.53	100.00
	50	93.51	83.55	100.00
	100	79.89	75.07	89.53
2	30	96.29	85.15	100.00
	50	98.87	96.06	100.00
	100	94.37	91.81	98.15
3	30	91.92	89.74	94.20
	50	90.19	86.53	92.04
	100	87.36	82.14	95.64

The results in the previous table show that the distance traveled using the decision tree show variable approximations to the results provided by the heuristic. The lowest accuracy in all scenarios was found with the largest number of orders, O = 100 orders. Although distance comparison presents moderate results in some instances (lowest average percentage of 79.89%), we also need to compare the results attained with the decision trees versus naïve heuristics commonly employed in warehouse real-life environments considered to be related to the scope of this study.

Batches of orders formed with the decision tree can possibly end in a low replicating percentage compared with FF-EBB heuristic distances, however, this low heuristic replication may result in significant savings against naïve heuristics such as FCFS and single picking. Therefore, we present a graph for each warehouse scenario comparing total traveled distances obtained with decision tree, FF-EBB, and FCFS heuristics. The comparison for warehouse scenario 1 is presented in Figure 4.

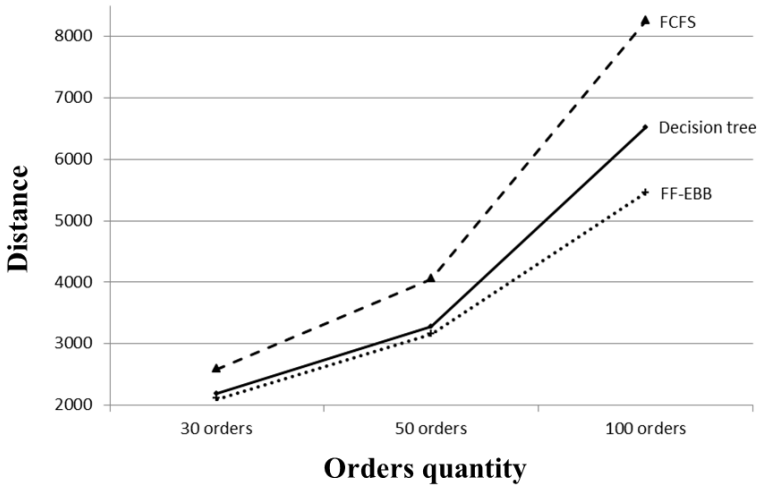

Figure 6. Decision tree, FF-EBB, and FCFS traveled distance comparison for warehouse scenario 1.

From the graph, it is clearly detected that the decision tree achieves satisfactory results with a small to medium amount of orders, but performs at its weakest when the number of orders is increased, however, in all cases the distance savings using the decision tree are significant compared to FCFS and single picking (the tree used to form batches for this scenario was presented in Figure 3).

Next, the results for scenario 2 are presented in Figure 5 and Figure 6:

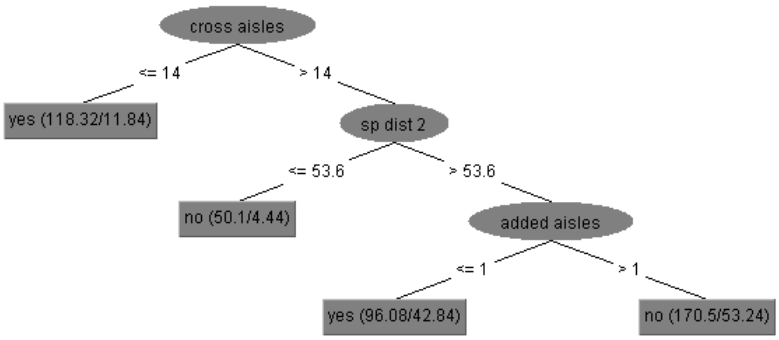

Figure 7. Decision tree mined from warehouse scenario 2 batches information.

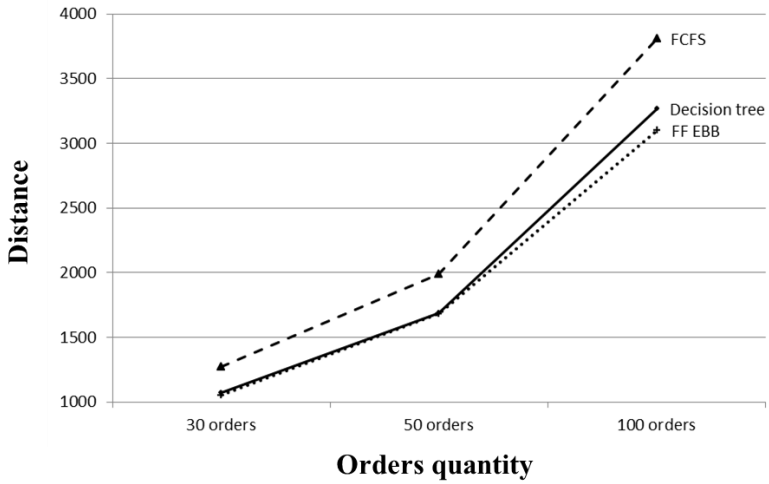

Figure 8. Decision tree, FF-EBB, and FCFS traveled distance comparison for warehouse scenario 2.

The tree in Figure 5 presents a new attribute than the previous tree; that is, *cross aisles*. As the number of aisles is increased in the scenario 2 layout (10 aisles), any tour reaching the tenth aisle would require to travel nine cross aisles one-way and the same aisles to return to the depot. Therefore, the first leave of the tree *cross aisles* ≥ *14* refers that batching the pair of orders result in a tour up to the seventh aisle or shorter in the warehouse. The rules utilized to form the order batches are presented:

IF cross aisles ≤ 14 THEN batch = yes

IF cross aisles > 14 AND sp dist 2 > 53.6 AND added aisles ≤ 1 THEN batch = yes

Otherwise batch = no

The results indicate a similar trend to those in the previous scenario (considerable distance savings compared to FCFS and single picking and weakest performance with large order quantities).

Results obtained in the third scenario are presented as follows,

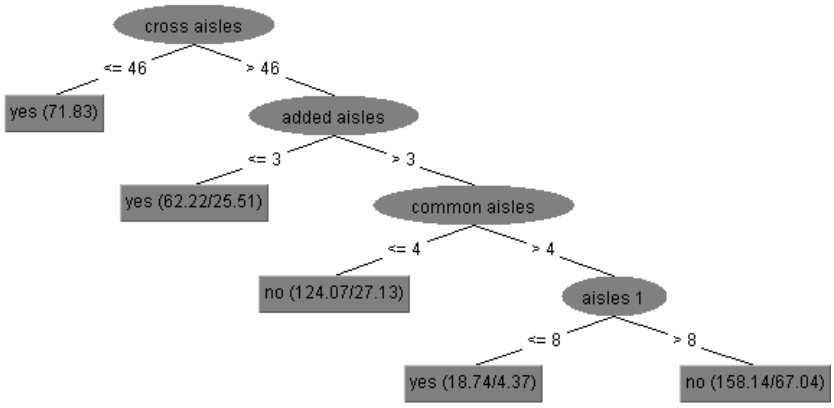

Figure 9. Decision tree extracted from warehouse scenario 3 batches information.

Figure 10. Decision tree, FF-EBB, and FCFS traveled distance comparison for warehouse scenario 3.

The decision rules mined from the third scenario information depicted in the decision tree in Figure 7 are presented as follows:

IF cross aisles ≤ 46 THEN batch = yes

IF cross aisles > 46 AND added aisles ≤ 3 THEN batch = yes

IF cross aisles > 46 AND added aisles > 3 AND common aisles > 4 AND aisles 1 ≤ 8 THEN batch = yes

Otherwise batch = no

A pattern consistent with previous scenarios is detected; the traveled distance batching the orders with the decision tree is approximated to the heuristic results although providing best results than the rest of the heuristics used for comparison.

In addition to compare the distance measure solely for analysis purposes, there exist additional comparison measures interesting in the warehouse operation, e.g. traveled aisles, picking facility efficiency, number of batches formed, and the like. Thus, a table including these significant measures is presented in Table 14.

Table 14. Overall results comparison for every warehouse scenario and order size.

	Order quantity	Batch method	Travel distance	Aisles	Cross aisles	Facility efficiency	Batches	Methods compared	Traveled distance	Aisles	Cross aisles	Facility efficiency	Batches
Warehouse Layout Scenario 1	30 Orders	DT	2193	35.75	63	0.93	11.75						
		FF EBB	2096	34.5	61	0.91	12	DT/FF-EBB	1.05	1.04	1.04	**1.02**	**0.98**
		FCFS	2587	43	73.5	0.82	13.5	DT/FCFS	**0.85**	**0.83**	**0.86**	**1.14**	**0.88**
		SP	3569	59	130.5	0.36	30	DT/SP	**0.61**	**0.60**	**0.48**	**2.59**	**0.39**
	50 Orders	DT	3281	55	100.5	0.95	19						
		FF EBB	3154	52.25	98	0.91	19.75	DT/FF-EBB	1.05	1.06	1.03	**1.04**	**0.97**
		FCFS	4053	68.25	121	0.84	21.5	DT/FCFS	**0.81**	**0.80**	**0.83**	**1.13**	**0.89**
		SP	6005	98	220	0.36	50	DT/SP	**0.55**	**0.56**	**0.46**	**2.64**	**0.38**
	100 Orders	DT	6519	106.5	199	0.98	36						
		FF EBB	5446	98.75	185	0.96	36.75	DT/FF-EBB	1.20	1.08	1.08	**1.02**	**0.98**
		FCFS	8241	138.8	236.5	0.85	41.5	DT/FCFS	**0.79**	**0.77**	**0.84**	**1.15**	**0.87**
		SP	12022	197.3	450.5	0.35	100	DT/SP	**0.54**	**0.54**	**0.44**	**2.79**	**0.36**
Warehouse Layout Scenario 2	30 Orders	DT	1074	69	153.5	0.89	8.75						
		FF EBB	1055	69.25	145	0.91	8.5	DT/FF-EBB	1.02	1.00	1.06	0.97	**1.03**
		FCFS	1270	83.25	177	0.78	10	DT/FCFS	**0.85**	**0.83**	**0.87**	**1.15**	**0.88**
		SP	2502	136.5	469.5	0.26	30	DT/SP	**0.43**	**0.51**	**0.33**	**3.44**	**0.29**
	50 Orders	DT	1685	110	233	0.95	13.25						
		FF EBB	1682	110.3	224	0.94	13.5	DT/FF-EBB	1.00	1.00	1.04	**1.02**	**0.98**
		FCFS	1991	131.8	265.5	0.84	15	DT/FCFS	**0.85**	**0.84**	**0.88**	**1.13**	**0.88**
		SP	4192	228	768	0.25	50	DT/SP	**0.40**	**0.48**	**0.30**	**3.79**	**0.27**
	100 Orders	DT	3267	214.8	450	0.95	26.25						
		FF EBB	3098	203	419	0.97	25.75	DT/FF-EBB	1.06	1.06	1.07	**0.98**	1.02
		FCFS	3807	255	510.5	0.87	28.75	DT/FCFS	**0.86**	**0.84**	**0.88**	**1.10**	**0.91**
		SP	8267	450	1551	0.25	100	DT/SP	**0.40**	**0.48**	**0.29**	**3.81**	**0.26**
Warehouse Layout Scenario 3	30 Orders	DT	4928	82	162	0.72	4						
		FF EBB	4506	74	142	0.96	3	DT/FF-EBB	1.09	1.11	1.14	0.75	1.33
		FCFS	5544	90	192	0.72	4	DT/FCFS	**0.89**	**0.91**	**0.84**	1.00	1.00
		SP	23268	329	1364	0.10	30	DT/SP	**0.21**	**0.25**	**0.12**	**7.50**	**0.13**
	50 Orders	DT	7782	130	256	0.82	6						
		FF EBB	7176	119	232	0.99	5	DT/FF-EBB	1.08	1.09	1.10	0.83	1.20
		FCFS	8988	146	288	0.82	6	DT/FCFS	**0.87**	**0.89**	**0.89**	1.00	1.00
		SP	36950	521	2078	0.16	50	DT/SP	**0.21**	**0.25**	**0.12**	**4.99**	**0.12**
	100 Orders	DT	16422	273	526	0.95	11						
		FF EBB	15736	257	516	0.94	11	DT/FF-EBB	1.04	1.06	1.02	1.00	1.00
		FCFS	17328	281	576	0.87	12	DT/FCFS	**0.95**	**0.97**	**0.91**	**1.09**	**0.92**
		SP	80172	1136	4572	0.10	100	DT/SP	**0.20**	**0.24**	**0.12**	**9.10**	**0.11**

The left section of the table presents merely performance measures achieved with each heuristic in every single scenario, whereas the right section of the table lists comparisons of the decision tree batches with the rest of heuristics. Note that a value smaller than 1 (emphasized in bold) signifies a superior performance of the decision tree (except in facility efficiency where a higher efficiency is to be preferred) against that particular heuristic.

In some cases the decision tree resulted in superior performance than the FF-EBB heuristic (denoted with accuracy = 100%), this condition was achieved because the heuristic method sort the orders by the envelope number in addition to sort the orders in terms of decreasing volume in such way that the last orders in the list could be that of large volume, leading to suboptimal utilization of the order picking facility. In contrast, forming orders batches employing the decision tree rules do not sort the orders in any way (the orders are compared upon arrival sequence), thus facility efficiency in later batches can be improved ensuing shorter picking routes.

4.2 Summary.

We developed a framework addressing the issue of whether order can be batched in a real-time warehouse environment following simple decision rules involving few meaningful attributes.

First, we batched a generated dataset using an existent heuristic simulating effective batch decisions. We construct a flat file comparing every possible pair of orders specifying the target concept as whether the orders were batched or not. A data imbalance

issue was encountered and resolved including cost-sensitive learning in our learning algorithm.

A decision tree learning algorithm was then applied to mine knowledge directly from the flat file, that is, without prior knowledge from the underlying method employed to batch the orders. The tree was then applied as a decision-making tool applicable to form batches of orders dynamically (real-time environment). Additionally, meaningful insights regarding why two orders are batched were drawn from the tree itself analyzing attributes utilized as tree leaves.

CHAPTER 5. CONCLUSION AND FUTURE WORK.

We developed a novel approach to a problem commonly confronted in warehouse environments: discover a practical application assisting in dynamic batching decision process, that is, in a real-time warehouse environment. In the framework, we applied data mining directly in historical order batches information, then translate this knowledge into simple rules expediting the batch decision process, even if no prior knowledge of the system is made available.

The framework proposed application is novel in the context of logistic operations (particularly to batch orders in a warehouse environment), found to build effective models to construct batches of orders in real-time warehouse scenarios. Moreover, the framework provides insights regarding why the batches are constructed, revealing the most meaningful attributes leading to batch a pair of orders.

Subsequent to several numeric experiments, we can define that batching orders following decision tree rules extracted from historical order batches information can lead to significant distance savings compared to simple heuristics commonly employed in warehouse environments because of its simplicity (such as FCFS and single picking). In warehouses using the heuristics before mentioned, even sub-optimal improvements in picking operations may be intensified in correspondence to the operation magnitude, the high human-labor involved and its continuous repeatability.

Particular routes to further extend the present study are devised. Foremost, in our study we reviewed one heuristic as the underlying structure utilized to form the order batches to learn from, however, real-world warehouses can be particularly concerned in fast delivery responses rather than picking distance efficiency or a trade-off between the two metrics. Hence, the framework may be extended analyzing different batching heuristics. Also, it would be interesting to review the framework performance using real warehouse and orders information; we infer that potentially beneficial insights would be gained with that approach.

REFERENCES.

Ackerman, K.B. (1997). Practical handbook of warehousing. Materials Management/Logistics series. 4th edition. Springer.

Bozer, YA; Kile, JW. (2008). Order batching in walk-and-pick order picking systems. International journal of production research, 46 (7) 1887-1909.

Chen, MC; Wu, HP. (2005). An association-based clustering approach to order batching considering customer demand patterns. OMEGA-International journal of Management Science 33 (4), 333-343.

Cormier, G., Gunn, E.A. (1992). A review of warehouse models. European Journal of Operations Research. Vol. 58. Issue 1. 3-13.

Coyle, J.J., Bardi, E.J., Langley, C.J. (1996) The management of business logistics. 6th edition. West Publishing Company. St Paul.

De Koster, M.B.M., Van der Poort, E.S., Wolters, M. (1999). Efficient orderbatching methods in warehouses. International journal of Production research 37 (7), 1479-1504.

De Koster, R; Le-Duc, T; Roodbergen, KJ. (2007). Design and control of warehouse order picking: A literature review. European Journal of Operation research 182 (2), 481-501.

Dekker, R., De Koster, M.B.M., Roodbergen, K.J., Van Kalleveen, H. (2004). Improving order-picking response time at Ankor's warehouse. INFORMS Vol. 34, No. 4, 303-313.

Drury, J. (1988). Towards more efficient order picking. IMM Monograph No. 1, The Institute of Materials Management, Cranfield, UK.

Elsayed, E.A., Unal, O.I. (1989). Order batching algorithms and travel-time estimation for automated storage/retrieval systems. International Journal of Production Research. Vol 27, No. 7, 1097-1114.

Gademann, N., Van de velde, S. (2005). Order batching to minimize total travel time in a parallel-aisle warehouse. IIE Transactions 37 (1), 63-75.

Gibson, D.R. and Sharp, G.P. (1992). Order batching procedures. European Journal of Operational research 58 (1), 57-67.

Gu, J.X., Goetschalckx, M., McGinnis, L.F. (2010). Research on warehouse design and performance evaluation: A comprehensive review. European Journal of Operational research 203 (3), 539-549.

Gu, J.X., Goetschalckx, M., McGinnis, L.F. (2007). Research on warehouse operation: A comprehensive review. European Journal of Operational research 177 (1), 1-21.

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H. (2009). The WEKA data mining software: An update. SIGKDD Explorations. Volume 11, Issue 1.

Hall, R.W. (1993). Distance approximation for routing manual pickers in a warehouse. IIE Transactions 25, 77-87.

Han, J. and Kamber, M. (2006). Data mining: Concepts and Techniques. Second edition, Elsevier Inc. ISBN 1-5586-0901-6.

Kotsiantis, S., Kanellopoulos, D., Pintelas, P. (2006). Handling imbalanced datasets: A review. GESTS International Transactions on Computer Science and Engineering. Vol 30.

Li, X. and Olafsson, S. (2005). Discovering dispatching rules using data mining. Journal of Scheduling 8, 515-527.

Olafsson, S., Li, X., Wu, S. (2008). Operations research and data mining. European journal of Operation Research 187, 1429-1448.

Pan, C.H., Liu, S.Y. (1995). A comparative study of order batching algorithms. OMEGA-International Journal of Management Science 23 (6), 691-700.

Quinlan, J.R. (1993). C4.5: Programs for Machine Learning. Morgan Kaufmann, San Mateo, CA.

Ratliff, H.D., Rosenthal, A.S. (1983). Order picking in a rectangular warehouse: A solvable case of the traveling salesman problem. Operations Research Vol 31, No. 3.

Rosenwein, M.B. (1996). A comparison of heuristics for the problem of batching orders for warehouse selection. International journal of Production Research 34 (3), 657-664.

Ruben, R. and Jacobs, F.R. (1999). Batch construction heuristics and storage assignment strategies for walk/ride and pick systems. Management Science. Vol 45, No 4.

Tompkins, J.A., White, J.A., Bozer, Y.A., Frazelle, E.H., Tanchoco, J.M.A. (2003). Facilities planning. John Wiley & Sons, NJ.

Tsai, C.Y., Liou, J.J.H., Huang, T.M. (2008). Using a multiple-GA method to solve the batch picking problem: considering travel distance and order due time. International Journal od Production Research 46 (22), 6533-6555.

Van den Berg, J.P. (1999). A literature survey on planning and control of warehousing systems. IIE Transactions 31 (8), 751-762.

Ye, Nong. (2003). The handbook of data mining. Lawrence Erlbaum Associates. ISBN 0-8058-4081-8.

ACKNOWLEDGMENTS.

I want to express my deepest gratitude to my wife, who was my anchor and at the same time my motivation during the whole research process.

My profoundest thanks to my family for shaping me with a progressive state of mind, and for their unconditional support.

My recognition to Dr. Sigurdur Olafsson for providing me with insightful comments and guiding me through the research process.

To the members of my committee, Dr. Jo Min and Dr. Suzuki Yoshinori, my special appreciations for your inspiring teaching style which helped in developing a mindset full of new thoughts and ideas.