MASTER

Route and Performance Analysis of Multi-Item Processes in Warehouse Automation Systems

Homvanish, Sivaporn

Award date:
2020

Department of Mathematics and Computer Science
Process Analytics Charter

Route and Performance Analysis of Multi-Item Processes in Warehouse Automation Systems

Master Thesis

Sivaporn Homvanish

Supervisors:
Dr. Dirk Fahland
Ir. Eefje van den Dungen
Ir. Hilda Bernard
Ir. Koen Verhaegh

Eindhoven, August 2020

Abstract

Process mining and novel techniques for route and performance analysis have been developed and applied on logistics processes for business services advancement. The existing techniques used with single-item processes consist of data complexity reduction, seeking significant locations by using an unsupervised-learning technique, and clustering route by the behaviors with the significant locations. However, the deployment of these techniques has been limited to single-item processes. The applicability in a multi-item process that encompasses multiple items and interrelation has not been validated. The main objective of this research is, hence, to adopt the existing process mining methods and techniques, which were originally developed for single-item process, and extend them to the process that involves multiple items in Warehouse Automation Systems (WAS).

Specifically, conceptualization of WAS, data transformation using the artifact-centric process mining approach, and the methodology for multi-item data pre-processing are applied, gaining the input for existing single-item techniques that perform route and performance analysis. Results of route clustering and performance assessment are visualized and interpreted, together with the verification by a domain expert, allowing the investigation of suitability in applying single-item techniques in WAS.

The results show that extending single-item techniques in WAS context can show the overall process highlighting the relationship between items. However, the data-driven approach without established ground-truth and the special characteristics of WAS domain negatively influence the interpretability of the outcome, questioning the applicability. Additional contextual information, more fine-grained analysis, and improving visualization can benefit future studies.

Keywords: process mining, multi-item process, lifecycle model, artifact-centric, multi-dimensional event log, warehouse automation system, route analysis, performance analysis, log extraction specification and pre-processing

Acknowledgement

This master thesis becomes reality with the kind help from many people. I would like to express my special thanks to all of them.

I would like to express my sincere gratitude first and foremost to my supervisor, Dirk Fahland for his constant and thoughtful guidance, valuable feedback and kind supports through the entire project. It was a great opportunity to work and study under his supervision.

In addition, I would like to thank wonderful people in Vanderlande Industries B.V. for their kindness and assistance. Special thanks to Eefje van den Dungen, Hilda Bernard, and Koen Verhaegh, who always helped, supported and supervised my progress. Then, I would like to thank Gijs van der Sanden and Jaimini M. A. Boender for their companionship throughout our graduation projects.

I am extremely grateful to my family and beloved for the encouragement, care and support which helped me in completion of this master thesis.

I would also thank to all of my BDMA friends for helping me during our studies, sharing good moments and having fun together throughout our two-year journey.

Last, I would not have the honor of being a student of Universit Libre de Bruxelles(ULB), Universitat Politcnica de Catalunya (UPC), and Eindhoven University of Technology (TU/e) without the generous Erasmus Mundus Joint Master Degree scholarship of European Commission. I would like to thank all BDMA committees for giving me this wonderful opportunity and priceless experiences.

Sivaporn Homvanish
Eindhoven, August 2020.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

This master thesis is the result of graduation project conducted at Vanderlande Industries B.V. as part of the Master of Computer Science in Erasmus Mundus Big Data Management and Analytics, coordinated by the consortium of Universit Libre de Bruxelles (ULB), Universitat Politcnica de Catalunya (UPC), Eindhoven University of Technology (TU/e), Technische Universität Berlin (TUB) and Université Paris-Saclay (CentraleSupelec). This graduation project has been supervised in Process Analytics research group from TU/e.

In this chapter, the motivation and the context of the study are explained in Section 1.1 Next, the research question is formulated and described in Section 1.2. The methodology to resolve the research question is introduced in Section 1.4. Then, we briefly explain about use case applied with this study in Section 1.3.

1.1 Motivation and Context

Many novel process mining techniques have been applied to automation of logistics processes in airports, parcel markets and warehouses in order to advance business services. The state-of-the-art methods and techniques work well in identifying physical product routes, clustering them based on significant physical locations and detecting the deviation from the optimum. More information about the state-of-the-art methods and techniques can be found in [3].

One of the main reasons of route clustering is the enormous size of the data and complexity. Full physical route data in the system can be larger than 10,000 of segments as pairs of steps over 10,000s of traces which leads to a very large feature space having more features than data points. With this large number of data, it also creates difficulties for analysts and domain experts to gain useful insights from the data.

However, the applicability of the state-of-the-art methods is limited to a single type of item and they do not allow analyzing the dynamics and interrelation between multiple items. This limitation hinders the analysis for Warehouse Automation Systems wherein items of multiple types are involved. In particular, the system may receive multiple pallets of products from suppliers, which are then unbundled and distributed into smaller trays. The trays are then kept in storages or batched with other trays into roll cages based on the received orders. Subsequently, the roll cages are delivered to the end customers.

In addition, the artifact-centric process mining [2] has been developed as an approach to analyze multi-item processes. With this approach, business process could be explored providing understandable each item's characteristic in the process and the interaction between different items. Hence, it could facilitate data and process understanding.

1.2 Research Question

Established techniques [3] have been implemented for single-item processes such as airport system to enable route and performance analysis at Vanderlande, but none has been applied to multi-item process like warehouse automation system. From a business perspective, implementing a new technique that has not been proven could come with costs in time and resources. The issue leads to our motivation to explore the possibility to adopt existing concept and techniques of single-item process that have been previously proven in other domains and apply in our context. We, thus, define the general research question as;

Given a dataset of Warehouse Automation System (WAS) and single-item techniques for route and performance analysis, could we adopt and extend the existing techniques to multi-item process to be able to understand the characteristics and performance of inter-related routes?

It is assumed that the data contains a timestamp, identifiers indicating each item and events considered as an activity could represent the whole process in the WAS. Moreover, it is assumed that data is available and ready to be used in some sort of database and WAS documents are complete and adequate for understanding the data.

1.3 Case Study

This research is conducted in the context of Warehouse Automation System, which had been implemented by Vanderlande Industry B.V. (referred to hereafter as Vanderlande). The applicability in warehouse domain remained unexplored and becomes the goal of this research.

Vanderlande is a business-to-business company that is active in logistic industry offering automation of logistics processes in airports, parcel markets and warehouses [4]. To have more practical and efficient operation, there is a demand to improve logistics processes but the improvement is confronting with challenges of the growth in size and complexity of solutions demanding the advancement of business service. Under collaboration with TU/e, process mining is widely adopted and improved at the company via the development of various techniques for the analysis of route and performance.

The warehouse system involves multiple items including pallets, trays, and roll cages. During operation, items are received and transferred from one to another based on a customer's order, in automation manner under the management through software to ensure the orchestration among many machinery modules creating data complexity where events of the same item is distributed over many tables.

In Chapter 3, warehouse process and data structure are explained by examples. Then, the proposed solutions are explained as step-by-step approaches which help us solve the challenges of our study in Chapter 4.

1.4 Methodology

Here, Cross Industry Standard Process for Data Mining (referred to hereafter as CRISP-DM) is followed as depicted in 1.1. In this study, we applied 5 steps on CRISP-DM; business understanding, data understanding, data preparation, modeling and evaluation.

1. The first step, business understanding, is done by reading technical specification, process documents, and discussing with domain experts to gain more understanding how the warehouse process works. Then, an artifact and its lifecycle in the system are defined allowing depiction of interrelation between artifacts. Explanation about business process is elaborated in Chapter 3.

2. The second step, data understanding, is obtained by checking database and data structure together with the procedure of machine message

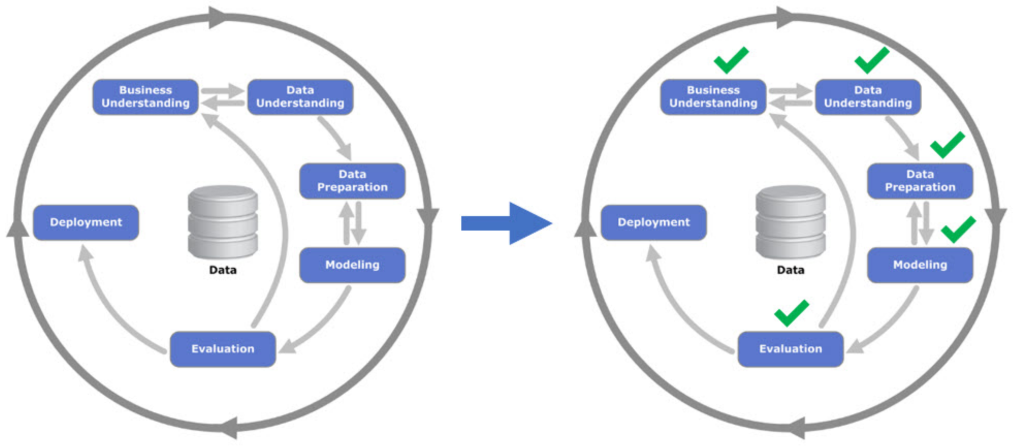


Figure 1.1: Diagram of CRISP-DM [1] and our approach

recording in WAS which is also mentioned regarding to data context in Chapter 3.

From the first and second steps, we can then define sub-research problems for data preparation, modeling, and evaluation in Chapter 4.

3. The third step, data preparation, is to pre-process data; in this case, raw data from many tables are transformed into event logs. In particular, main events of the process are first identified and then categorized by traces in the process. Since data is retrieved from multiple sources where the consistency cannot be guaranteed, therefore it is necessary to clean data by removal of outlier and out-of-scope items and by data verification. Detailed explanation is described in Chapter 5.

4. The forth step, modeling, is to integrate the derived event-logs with single-item process analyses on route and performance explained in Chapter 6. Then, we visualize the clusters and routes after integrating our event logs with single-item techniques and also use textual description enabling further analysis which is detailed in Chapter 7.

5. The fifth step, evaluation, is performed to verify the soundness of the visualization and textual description obtained from the methods; it facilitates the assessment of the applicability of the existing methods to multi-item process. More information is described in Chapter 8.

Chapter 2

Preliminaries

In this chapter, we explain all necessary fundamental concepts to provide background knowledge, terms, techniques used in this thesis.

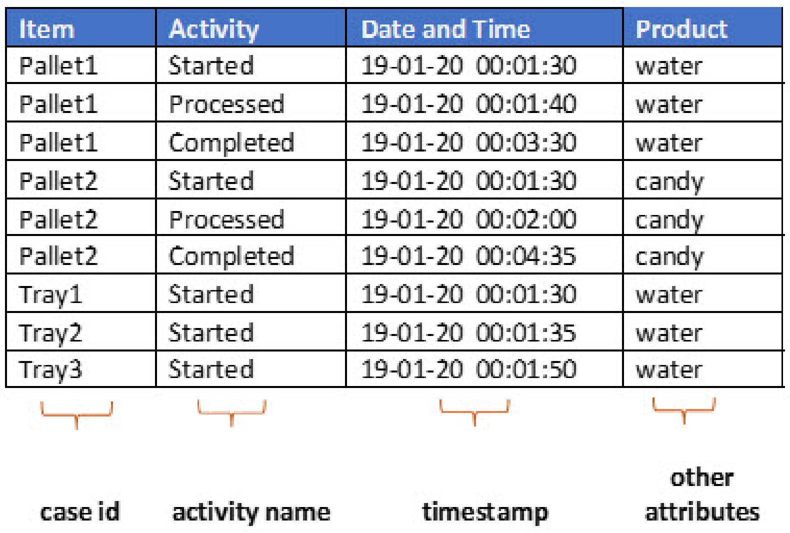
Three main topics are referred. We first explain about process mining by mainly presenting event log in Section 2.1. Then, we provide explanation about multi-item process, single-item process, their difference and the process mining method used for multi-item data pre-processing in Section 2.2. Lastly, Section 2.3 provides fundamental idea of existing techniques for summarizing routes of single-item processes.

2.1 Process Mining

Process mining is a technique for extracting knowledge from the event logs in present-day information systems. It is an interdisciplinary research blending data mining and computational intelligence and can be generally classified into three categories; process discovery, conformance checking and process enhancement [5]. In this study, we mainly focus on how to transform warehouse data to an appropriate format, so called event log, that can be used in process mining.

Event log is normally used as an input for process mining techniques. Prime examples include database, transaction log, and audit trail and spreadsheet, which are transformed data that records behavior of a process. Three attributes are required for each event in an event log: case, activity and timestamp. However, an event log could have another attribute that provides additional information of the event, for instance, event executor, and resource utilization [6] [7].

Table 2.1 shows a simplified example of an event log in a warehouse process with the main component and its attributes. Each row in the table represents one event and the events are grouped per case in this example. Columns “Item”, “Activity” and “Date and Time” serve as attributes case, activity and timestamp respectively, satisfying the minimal requirement of an event. The last column “Product” is an example of supplementary attribute that could be recorded in the event log.



Item	Activity	Date and Time	Product
Pallet1	Started	19-01-20 00:01:30	water
Pallet1	Processed	19-01-20 00:01:40	water
Pallet1	Completed	19-01-20 00:03:30	water
Pallet2	Started	19-01-20 00:01:30	candy
Pallet2	Processed	19-01-20 00:02:00	candy
Pallet2	Completed	19-01-20 00:04:35	candy
Tray1	Started	19-01-20 00:01:30	water
Tray2	Started	19-01-20 00:01:35	water
Tray3	Started	19-01-20 00:01:50	water

Table 2.1: Example of Event Log

Moreover, there are 2 definitions normally used in process mining which are case and trace.

- Case represents physical items handled in the process.
- Trace represents sequence of events of a case.

From the example in Table 2.1, there are 5 cases; two cases for pallet which are Pallet1 and Pallet2 and other three cases Tray1, Tray2, Tray3 for tray. Both Pallet1 and Pallet2 have a same trace <Started, Processed, Completed>, while Tray1, Tray2 and Tray3 have a trace as <Started>.

2.2 Multi-item Process vs Single-item Process

Multi-item process is a process that encompasses a vast variety of type of items. It differs from single-item process that has only one item in the process, which could be traced from start to end through the whole process.

2.2.1 The Differences between Multi-item Process and Single-item Process

Multi-item process is dissimilar to single-item process with four main differences.

1. The number of items in the process;

   A single-item process has only one item, while multi-item process has more than one items in the process.

2. Lifecycle of each item;

   Different items in multi-item process have their own lifecycles that represent the different major activities within the same range of time and synchronize with the lifecycle of other items at certain steps. Meanwhile, the single item in single-item process contains only one lifecycle while it is applied for every item in the process.

   Moreover, all records of all items having same type in a single-item process are logged in an order. In other words, we could obtain only one item when identifying events by a specific time in the data. On the other hand, the behavior of multi-item process is no longer a sequence of event but multiple related sequences depending on items, resembling a graph that could have one node linking multiple nodes or an order that contains multiple partial orders inside.

3. Relation between items;

   As single-item process has only one item, hence, there is no relation to be considered in this process. In contrast, multi-item process has handover activities creating a relation between different items. The relation could be one-to-one, one-to-many, or many-to-many relations.

4. The interaction between item lifecycles;

   We could observe the interaction between item lifecycles in a multi-item process. Hence, we could perform an analysis on how one item could impact another, which is not possible in a single-item process.

2.2.2 Artifact-Centric Process Mining

Artifact-Centric process mining is one of the multi-item process mining techniques. Its output is an artifact-Centric process model that describes processes with multiple items, items' lifecycles, relations and interactions. It could prevent false dependencies according to data divergence and convergence [8].

Figure 2.1 shows the overview on artifact-centric process discovery. The input of this technique could be either event log or relational database. Then, the input is used to create data model having multiple items with own case identifier and their relations, together with artifact-centric process model as outputs. In brief, the goal of this technique is to relate the recorded data with modeled behavior; it can be conducted by following this procedure [2]:

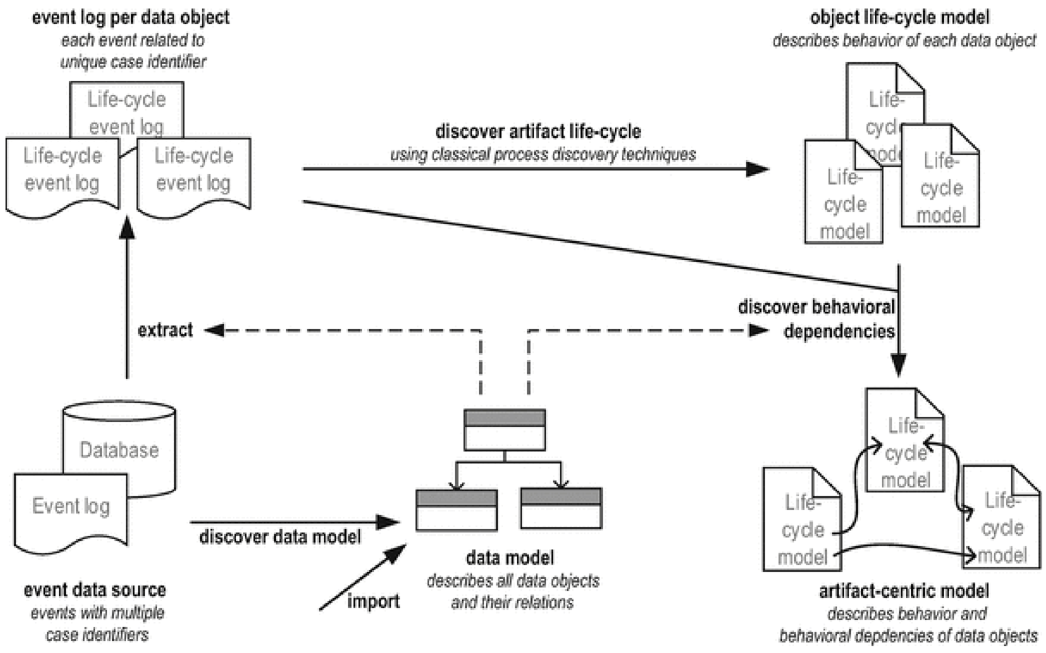

event log per data object
each event related to unique case identifier
Life-cycle event log
Life-cycle event log
Life-cycle event log
discover artifact life-cycle
using classical process discovery techniques
object life-cycle model
describes behavior of each data object
Life-cycle model
discover behavioral dependencies
extract
Database
Event log
discover data model
import
data model
describes all data objects and their relations
event data source
events with multiple case identifiers
artifact-centric model
describes behavior and behavioral depdencies of data objects

Figure 2.1: Overview on artifact-centric process discovery [2]

1. Identify items in the process;

   The first matter that we should consider is the number of items in the process. Therefore, all independent items have to be identified from the data.

2. Identify lifecycle of each item;

   After item discovery, it is crucial to understand the behavior of each item and identify its lifecycle. From this step, we could distinguish items within the same type and understand the key behavior of different item lifecycles.

3. Identify relation between items;

   The output yielded from the previous step has separate events of each item by item lifecycles. Hence, we need to define the relationship that we could relate items and link them together through the whole dataset.

4. Identify interaction between item lifecycles;

   To enable fruitful analysis, interaction between item lifecycle is included to find behavioral dependencies, which could be defined based on the interests in any kind of process. As a result, we could create a complete behavior model representing all items and their interactions.

2.3 Existing Techniques of Single-item Process for Route and Performance Analysis

In this section, we explain the state-of-art technique from prior master thesis [3] that are currently used with single-item process at Vanderlande. All methods in this section have been proven that they could determine route and assess performance of baggage handling system (BHS) and provide interesting insights.

We first introduce the definition of route and logistic step. Route is a sequence of activities/locations, different bags in BHS can follow the same route. Logistic step is a subsystem in BHS denoting one specific logistic task. As BHS has over 2,000 individual locations, all locations are organized into a set of subsystems having different logistic functions.

Figure 2.2 shows the overview of single-item technique, consisting of three steps. The result of each step will be an input of the next step.

The initial input of this technique is a set of route per logistic step in the system and the final output is route clusters.

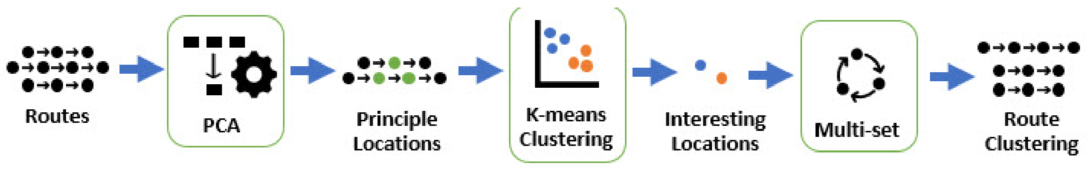
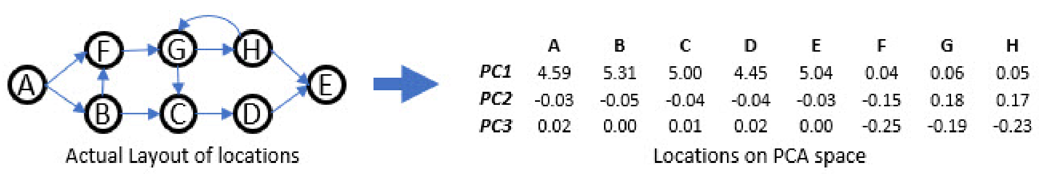


Figure 2.2: Overview of the Single-item Technique used in this study

Step 1: identify all activities/locations which are significantly related to different routes.

Principle Component Analysis (PCA) is applied to find those activities/locations. Then, it returns set of “principle locations”. However, set of principle locations is still too large because many very similar routes share some principle locations but also have additionally other principle locations that makes them different, although we would consider them similar.

Simplified example is illustrated in Figure 2.3. Eight locations – A,B,C,D,E,F,G,H – are projected on PCA space. Each location has it own values presenting the level of significantly related to different routes.



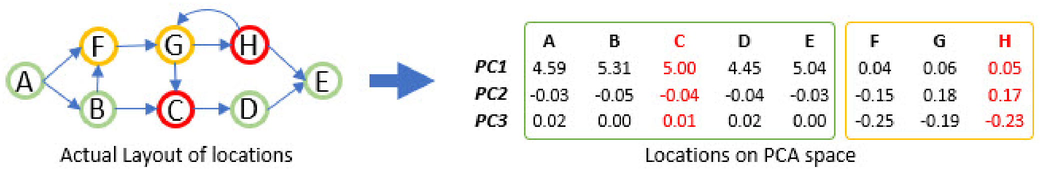
	A	B	C	D	E	F	G	H
PC1	4.59	5.31	5.00	4.45	5.04	0.04	0.06	0.05
PC2	-0.03	-0.05	-0.04	-0.04	-0.03	-0.15	0.18	0.17
PC3	0.02	0.00	0.01	0.02	0.00	-0.25	-0.19	-0.23

Figure 2.3: Simplified example of Single-item Technique Step 1

Step 2: reduce set of principle locations further to set of “interesting locations”.

K-means clustering is used in PCA space to reduce principle locations to a set of interesting location. The location which is the closet to the centroid of each cluster in PCA space are chosen as an interesting location.

After applying K-mean clustering on the simplified example in Figure 2.3. We could yield a result in Figure 2.4. The locations highlighted in the same colour indicate that they are in the same cluster. Hence, two clusters are identified; green and orange clusters. Assume that location C and H are closet to the centroid of green and orange clusters respectively. As a consequence, location C and H are chosen as interesting locations.



	A	B	C	D	E	F	G	H
PC1	4.59	5.31	5.00	4.45	5.04	0.04	0.06	0.05
PC2	-0.03	-0.05	-0.04	-0.04	-0.03	-0.15	0.18	0.17
PC3	0.02	0.00	0.01	0.02	0.00	-0.25	-0.19	-0.23

Figure 2.4: Simplified example of Single-item Technique Step 2

Step 3: group all routes which have the same multi-set of interesting locations into one route cluster.

Routes in one route cluster may show interesting locations in a different order, but if an interesting location occurs for example two times in a route, then all routes in this cluster visit this location twice

We use the result from K-means clustering in Figure 2.4 as an input of multi-set abstraction of interesting locations. The result could be obtained as shown in Figure 2.5. Bag 1 and Bag 4 visit interesting location C, thus they are in the same group, Cluster 1. Bag 2, Bag 3 and Bag 5 pass interesting location H once. These three bags are in Cluster 2. Meanwhile, Bag 6 also visit location H but it occur two times. Hence, this bag is clustered to the different group called Cluster 3.

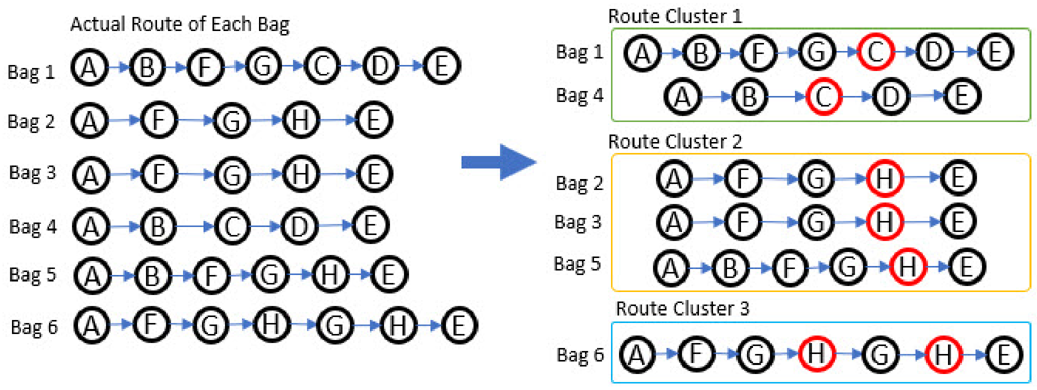


Figure 2.5: Simplified example of Single-item Technique Step 3

2.3.1 Principal Component Analysis

Principal Component Analysis (PCA) is a technique that is used to reduce the dimension of a large dataset by finding principle components that could represent most information of data, while reducing the number of variables [9]. This technique is implemented on each of logistic steps in BHS to reduce the huge number of routes which is complex and difficult to gain insights. By applying PCA and pertaining only principle components which are activities/locations, we could obtain a new compressed data that could still preserve much information of the data, easing exploration and visualization. Then, this compressed data is utilized in the next step to identify a set of principle locations in each logistic step deemed to be locations that should gain extra attention. These location might well represent the entire route at conceptual level. In other words, instead of examining all locations in a specific route in a particular logistic step, only interesting locations are kept as they could provide general representation of the whole route by themselves already.

To exemplify, a system with four locations has links A-B, A-C, B-D, and C-D, as depicted in Figure 2.6. Possible routes from location A to location

D are to go either via B (A-B-D) or via C (A-C-D). Apparently, B and C can be supposedly more important than A and D as these two locations can differentiate two possible routes thereby capturing more variants of routes, while every route passes A and D. If PCA manages to learn the importance of B and C, and then we select the principle component that explains this route variability (equivalent to keeping these two locations), the locations A and D, which are less important, can be discarded. This would result in a compact representation of data while preserving important characteristics.

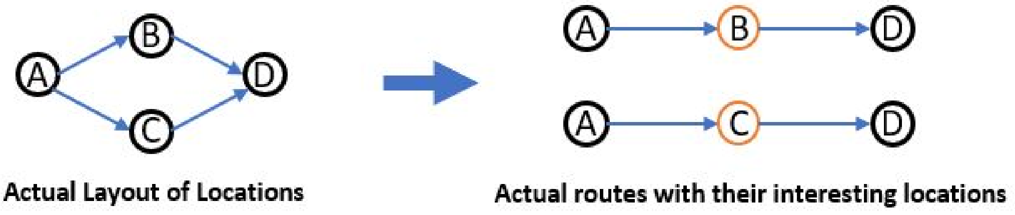
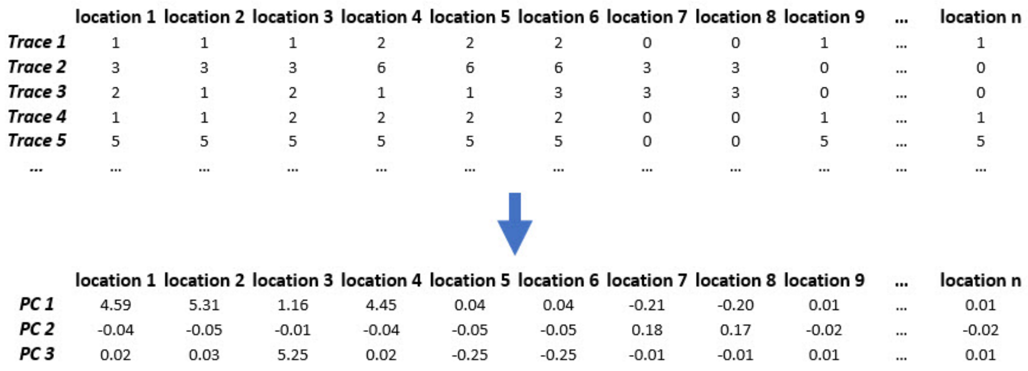

A
B
C
D
Actual Layout of Locations
A
B
D
A
C
D
Actual routes with their interesting locations

Figure 2.6: Example of actual layout of location and its routes

One important setup currently used with BHS is the number of dimension of PCA. As we mention that PCA is a dimension reduction technique. Hence, we could identify the number of PCA dimensions we would like to obtain as an output. The existing setup uses three dimension of PCA to conduct the analysis. As a consequence, we adopt this idea with our study. Figure 2.7 shows an example of three-dimension PCA we could gain as an output which we call compressed data.



	location 1	location 2	location 3	location 4	location 5	location 6	location 7	location 8	location 9	...	location n
Trace 1	1	1	1	2	2	2	0	0	1	...	1
Trace 2	3	3	3	6	6	6	3	3	0	...	0
Trace 3	2	1	2	1	1	3	3	3	0	...	0
Trace 4	1	1	2	2	2	2	0	0	1	...	1
Trace 5	5	5	5	5	5	5	0	0	5	...	5
...	...	...	...	...	...	...	...	...	...	...	...

	location 1	location 2	location 3	location 4	location 5	location 6	location 7	location 8	location 9	...	location n
PC 1	4.59	5.31	1.16	4.45	0.04	0.04	-0.21	-0.20	0.01	...	0.01
PC 2	-0.04	-0.05	-0.01	-0.04	-0.05	-0.05	0.18	0.17	-0.02	...	-0.02
PC 3	0.02	0.03	5.25	0.02	-0.25	-0.25	-0.01	-0.01	0.01	...	0.01

Figure 2.7: Input and Output of 3 dimension PCA

2.3.2 K-means Clustering

K-means clustering is an unsupervised techniques that is used to group unlabeled instances of data. The number of clusters is a user-defined parameter K [10].

The aim of using this method is to find an interesting location of each logistic step that could be a representative of a set of locations having similar behavior in a same cluster.

This technique is applied with the compressed data gained from PCA method in BHS. Each location in compressed data are located, in principle component (PC) space, close to another location that has similar characteristic, resulting in multiple groups of locations in the space. Then, K-means clustering is performed yielding the centroid of each cluster; a representative location of each cluster is selected as the instance which is closet to the centroid and supposed to be an attention-worth location, thereby named as interesting location.

Figure 2.8 illustrates the obtained result of one logistic step from applying K-means clustering with compressed data from three dimensions PCA. In this example, K is defined as four. Each colored dot on the graph represents an instance of location in the system and the coordinate x, y and z are the value of such instance projected to PC1, PC2, and PC3 respectively. Hence, we could see four different groups with different colors representing particular clusters and the dot in a red circle is the representative location of that cluster.

The currently-used approach at Vanderlande with BHS defines a single value of K for finding interesting locations in the system. In other words, all logistic steps in BHS have the same number of interesting locations.

Moreover, the interesting locations are used with the next technique named multi-set for clustering routes in the system.

2.3.3 Multi-set

Multi-set is one of data structure storing a collection of elements which allows duplication. It is dissimilar to a normal set whose elements in the collections can not be duplicated [11].

This concept is implemented in BHS for clustering routes. The interesting locations gained from K-means clustering is used for grouping routes that pass through the interesting locations. However, as multi-set allows duplication, it means that, by applying this concept, the frequency of a specific visited location is considered. Hence, if some routes pass through the same interesting locations with different number of visiting, all routes are

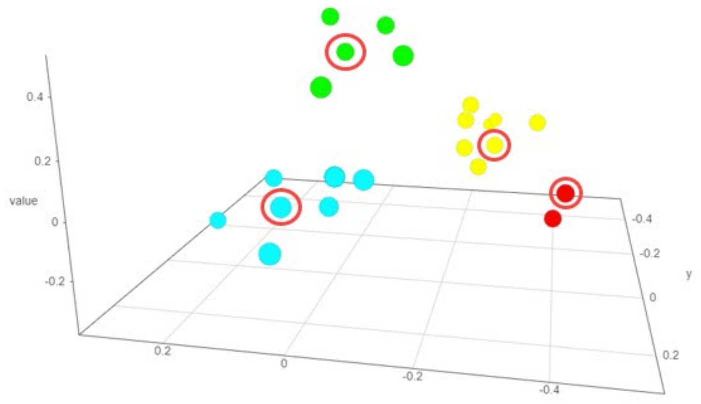


Figure 2.8: Example Result of K-means clustering with K = 4

not grouped in the same cluster but are separated in different groups having different number of visiting.

The main advantage of using multi-set in BHS allows the capability of detecting repeated activities or the presence of loop. Hence, the route passing through any interesting locations more than once could be detected.

2.3.4 Visualization

The current BHS visualization is implemented with the aim to depict clusters of routes, the number of baggage in each cluster providing start and end point of routes and the performance (execution time) of each baggage through different part of system.

Data is separated into routes per logistic step. Each logistic step has their own clusters and each cluster can belong to only one specific logistic step. Hence, the visualization shows sequence logistic steps as sequence of cluster having start and end locations.

All routes are visualized in form of graph. Two main components of graph are nodes and edges. Nodes represent locations (start and end locations in each path of the system) and clusters. Meanwhile, edges represent the trace of baggage via a sequence of locations and clusters.

Figure 2.9 shows the structure of the visualization. The color legend on the top of visualization shows ranges of performance measured as execution time in seconds. Below the legend is the visualization graph. Red box represents a logistic step which means that all nodes and edges in that box belong to the same logistic step. Each logistic step consists of start, end locations and clusters. Start and end nodes have darker blue color referring to the physical locations in the system, while the lighter blue nodes are logical nodes representing clusters. This visualization shows bags following the same sequence of logistic steps. The full visualization could be seen in Figure 2.10.

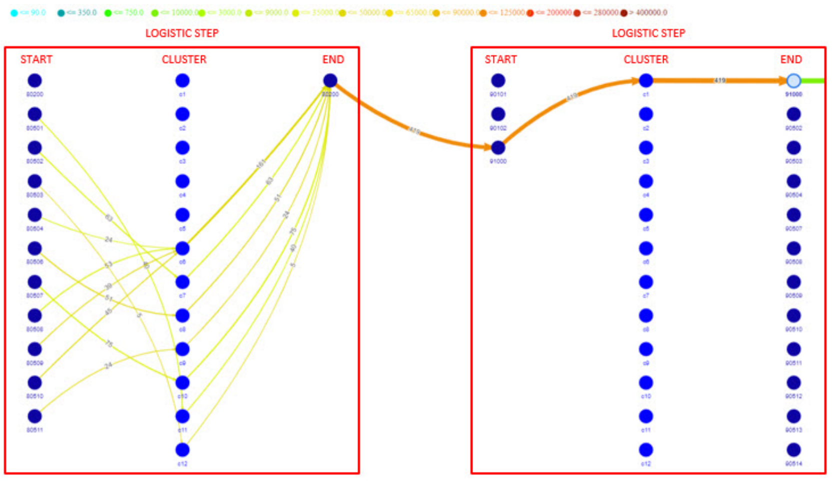


Figure 2.9: Visualization Structure

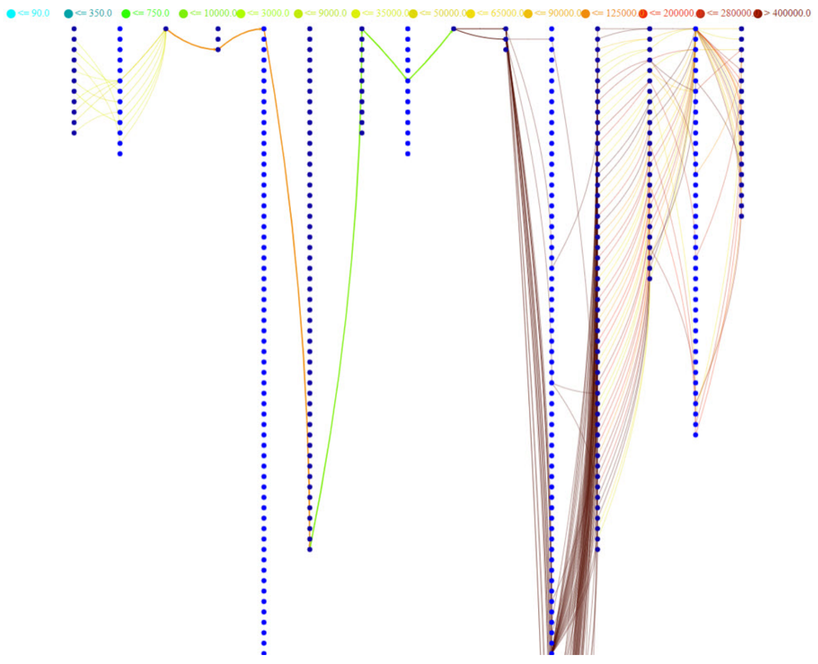


Figure 2.10: Visualization Structure

Chapter 3

Use Case

In this Chapter, Warehouse Automation System is introduced as an example of multi-item process with complex data structure.

Section 3.1 introduces the warehouse process that occurs through the entire system. In Section 3.2, we explain process lifecycle and hierarchy. Then, data context is described showing WAS data structure, its complexity, scope and limitation of the analysis in Section 3.3. We finally elaborate our research question and break down into multiple sub-questions in Section 3.4.

This chapter explains business understanding and data understanding steps of our methodology in CRISP-DM which are previously explained in Section 1.4.

3.1 Warehouse Automation System

Before using warehouse data for any analysis, it is important to understand how the process in WAS works, so that we can define item lifecycle and process hierarchy which are required for data pre-processing as a next step.

Section 3.1.1 outlines the warehouse process from the start to the end, depicting how the process works and providing more understanding on multi-item process behavior.

3.1.1 Warehouse Process

Warehouse can be defined as *"a large building for storing goods and products prior to distribution"* [12]. A warehouse receives products from various suppliers and fulfils orders of the end customer by consolidating a set of products that usually come from different pallets into one or more roll cages based on customer's requests. This process is qualified to be a multi-item process because there are 3 primary items involved in the process, which are pallet, tray, and roll cage. Figure 3.1 illustrates the end-to-end process consisting of 9 steps which are explained as follows.

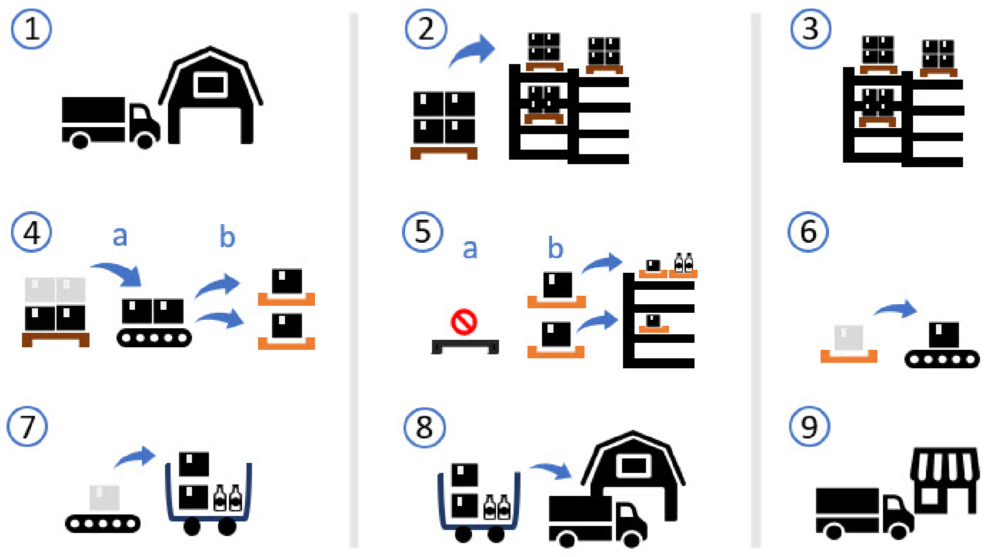


Figure 3.1: An Overview of Warehouse Process

1. A process starts when suppliers deliver pallets and the pallets enter inbound area in warehouse.

2. The pallets are forwarded to *Pallet Storage*.

3. Pallets are parked in the storage until replenishment order has arrived.

4. a. Pallets are transported from the storage to unbundle stations.

   b. Products on the pallets are distributed into multiple trays.

5. a. Pallets leave the system after finishing unbundling.

   b. Trays are forwarded to *Tray Storage*.

6. When customer orders arrive, trays are transferred to bundle station.

7. The products from multiple trays are stacked on single or multiple roll cages.

8. Roll cages are conveyed to docks and loaded on trucks.

9. Trucks deliver a set of products to end customers.

Apparently, only steps 1 to 7 take place in WAS, while steps 8 and 9 are not handled in this system and hence are not included in the scope of this study.

3.2 Process Lifecycle and Hierarchy

Multi-item process like warehouse process has a different behavior from single-item process. As we deal with multiple items in the system, it is crucial to understand the behavior of each item and its lifecycle, subsequently allowing us to find relations and interactions among items. This section provides the explanation of items, item lifecycles and end-to-end process flow occurring in WAS. In addition, the hierarchy of WAS is also explained to show the general concept that is used in the system. The numerical labels appearing in the figures in this section are in line with the warehouse process appeared in Figure 3.1.

As an exploratory research, we focus on pallet and tray lifecycle together with their interrelation, while roll cages are excluded from this study because they have similar behavior to pallets, and the implication of this study may, later on, apply to roll cages.

3.2.1 Lifecycle Model Concept

Pallet Lifecycle

Pallet is used at the beginning of the process once warehouse receives a pallet of products from suppliers until the products are distributed to multiple trays. Pallets can be classified into two different types: pool pallets, which are the pallets that come from suppliers having unique identification number, and slave pallets, which are pallets that are used internally in the warehouse and could be reused for multiple times. Figure 3.2 illustrates pallet lifecycle and the process is described as follows.

1. Starting from inbound area, pool pallets are stacked on the top of slave pallets to ensure the stability and endurance of pallet while being forwarded around the system with 1:1 relation. (1 in Fig. 3.2)

2. Products on both pool and slave pallets are stored in *Pallet Storage.* (2 in Fig. 3.2)

3. Pallets are kept in the storage. When replenishment order arrives, pallets are forwarded to *Unbundle station*. (3 in Fig. 3.2)

4. All products on each pallet are unbundled and distributed to multiple trays. (4a in Fig. 3.2)

5. After finishing unbundling, empty pool and slave pallets are de-stacked then leave the system. (5a in Fig. 3.2)

For the last step, empty slave pallets return to buffer and could be reused with another new-coming pool pallet. Even though it is specified that both pool and slave pallets leave the system, the true meaning is that all the activities after this step are not recorded from the system's perspective. Only slave pallets are re-used in the next cycle meaning that their activities will start being recorded again.

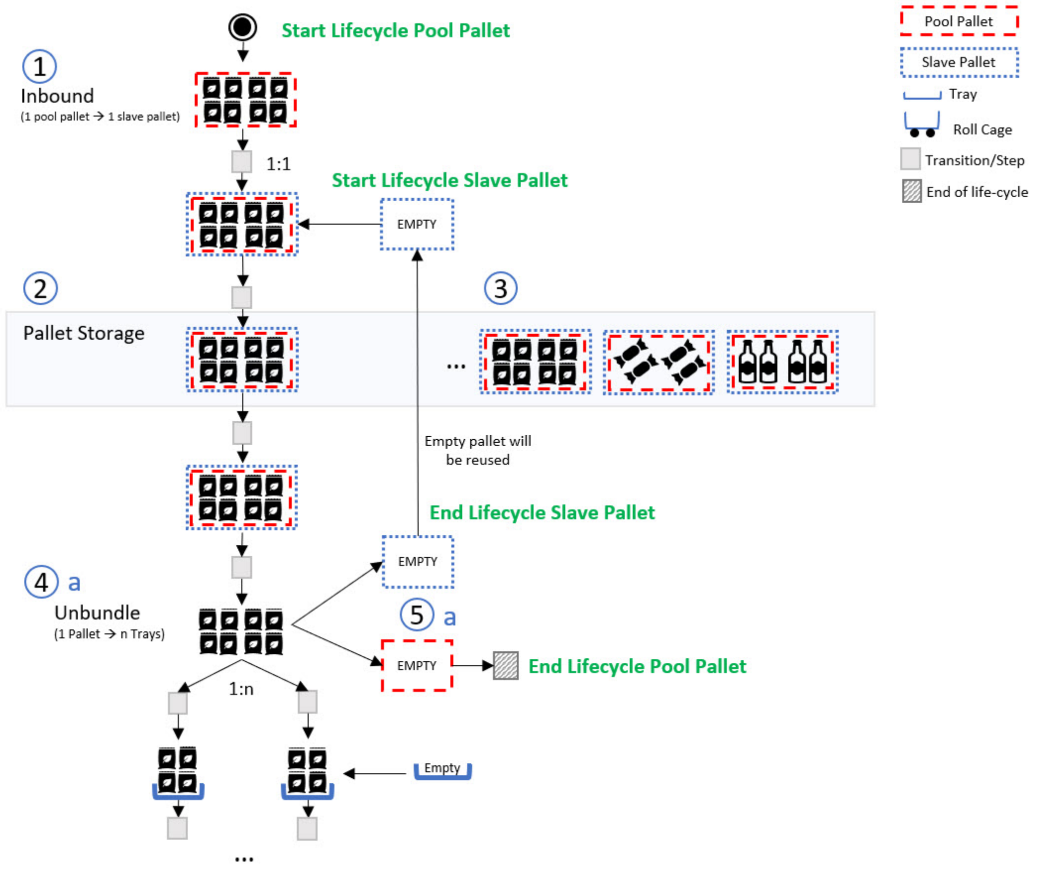

Start Lifecycle Pool Pallet
1
Inbound
(1 pool pallet → 1 slave pallet)
1:1
Start Lifecycle Slave Pallet
EMPTY
2
Pallet Storage
3
...
Empty pallet will be reused
End Lifecycle Slave Pallet
EMPTY
4 a
Unbundle
(1 Pallet → n Trays)
5 a
EMPTY
End Lifecycle Pool Pallet
1:n
Empty
...
Pool Pallet
Slave Pallet
Tray
Roll Cage
Transition/Step
End of life-cycle

Figure 3.2: Pallet Lifecycle Model

Tray Lifecycle

Tray is another important item in WAS that collects products from different pallets and combines them to a roll cage for delivering to end customers. This section explains how tray handles products until the point that the handled products are combined to multiple roll cages.

The behavior of trays in the system is different from the behavior of pallets. They can be reused for multiple times with two possible statuses 1) trays are empty 2) tray are still filled with products. For an empty tray, its lifecycle starts when products are put on the tray at *Unbundle station* and ends when the tray becomes empty. This cycle is called a full cycle. During each full cycle, trays are in the filled status and contain sub-cycle(s) depending on how often products are taken out from them into a roll-cage. However, a sub-cycle can start when products are moved to trays or when un-empty trays are stored in the *Tray storage*. Then, the cycle ends when products are moved from trays to roll cage(s).

Figure 3.3 shows the tray lifecycle and its full cycle and sub-cycle(s).

1. Starting from *Unbundle station*, products from pallets are distributed to multiple empty trays (4b in Fig. 3.3).

2. Two directions are possible after tray leaves *Unbundle station*: 1) trays are forwarded to *Bundle station* directly or 2) trays are stored in *Tray storage* and will be forwarded to *Bundle station* later when there is an incoming order from end customers (5b in Fig. 3.3).

3. In the case that trays are moved to *Bundle station* which is the start of sub-cycle, some products on the trays are distributed to designated roll cage(s). The relation between tray and roll cages could be 1:1 or 1:many. If trays are still not empty, they will be transported back to *Tray storage*. This step is considered as the end of sub-cycle (6 in Fig. 3.3).

4. Once there is no product left on trays, trays are turned to empty tray status and their lifecycles end (7 in Fig. 3.3).

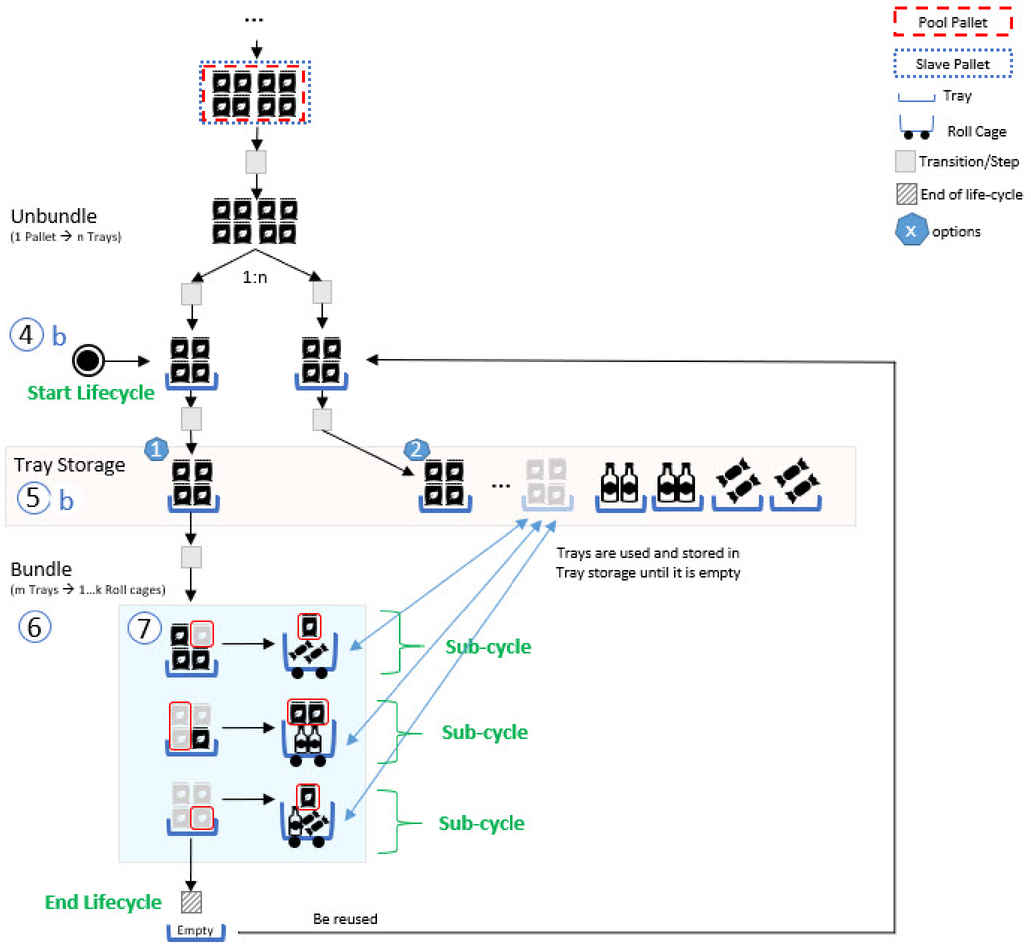


Figure 3.3: Tray Lifecycle Model

Complete Lifecycle

After lifecycles of pallet and tray are known, we could now find the relation between these two items and link them together. This section explains a complete lifecycle representing the relations among pallets, trays, and roll cages.

Figure 3.4 illustrates a complete cycle of WAS.

1. WAS receives pool pallets from suppliers at the inbound area and stacks each pool pallet on a slave pallet (1 in Fig. 3.4).
2. Products on stacked pool and slave pallets are transported to *Pallet Storage* (2 in Fig. 3.4).
3. Pallets are kept in *Pallet Storage* until receiving replenishment request (3 in Fig. 3.4).
4. Pallets are forwarded to *Unbundle station* and the products are distributed to multiple trays until pallets turn to be empty (4a, 4b in Fig. 3.4).
5. Empty pallets leave the system (5a in Fig. 3.4) while trays with products are directly moved to Bundle station or transferred to *Tray Storage* (5b in Fig. 3.4).
6. Once customer orders arrive, the requested trays are moved to Bundle station. This step is the start of sub-cycle (6 in Fig. 3.4).
7. Products on the selected trays are combined to a roll cage (7 in Fig. 3.4).

The connection among items in WAS could be seen in steps 4 and 7. Step 4 has the relation between pallet and trays having 1:many relations, while step 7 has the relation between tray and roll cages. This relation could be many:1 or many:many depending on customer order and the number of products on each tray.

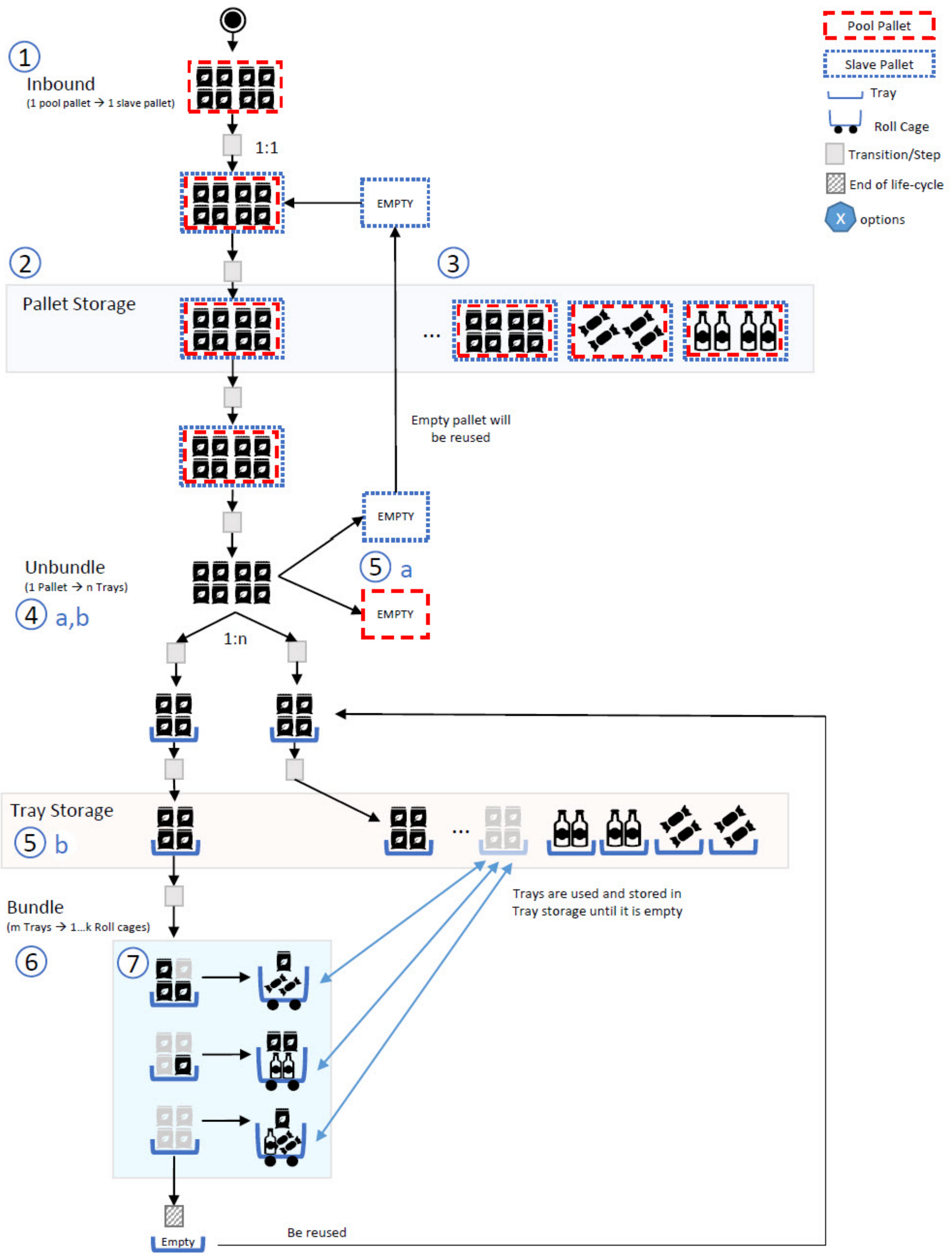


Figure 3.4: Complete Lifecycle Model

3.2.2 High Level Process

Lifecycles are defined from physical movements of items at different locations in warehouse, but these movements take place at different areas of the system which perform a specific logistic task, so called logistic step. Moreover, there are more than 1,000 physical locations in the warehouse. This creates difficulties for an analysis and gaining useful insight. Hence, we define high level (HL) steps to represent core activities by aggregating locations that fall in the same logistic step. Once we can define all HL steps, we could derive HL process that explains the steps in the warehouse at higher abstraction level.

It could be said that a HL step is simply a set of locations in a specific part of the system. As long as an item is at/moving between any location of the HL step, we consider that this HL step is being executed. This concept is adopted from BHS two-level structure, which is assumed for the route clustering technique to be applied in this study.

An example of the derived HL process is shown in Figure 3.5. The HL process starts at PALLET_INBOUND representing all activities once a pallet is entering inbound area. Afterward, the pallet moves to storage which is denoted as PALLET_STORAGE. Once, pallet is requested, PALLET_UNBUNDLE is recorded to show the steps conveying pallet from storage to Unbundle station. The relation between pallet and trays occurs when products are transferred from pallet to trays, as can be seen as the connection with the next HL step: TRAY_UNBUNDLE. This is followed by TRAY_STORAGE representing the movement of trays to storage, and lastly, TRAY_BUNDLE occurs when trays go to Bundle station and products are transferred to roll cage(s).

3.2.3 Low Level Process

A high level step explained in the previous section is composed of physical movement steps by locations that we call a logistic step, which describes the activities in the system at higher abstraction level. In contrast, a low level step is defined as a physical movement step or a routing step that specifies the actual locations that are transited by items in the system.

In other words, we use physical movement step for low-level and logistics step for high-level representation. Additionally, if we consider low-level routing locations, there are two elements to be considered: physical locations (at low level) and logistic step (at high level). As a sequence of activities, occurring at various locations, share the same high-level step, we can group all low-level steps in the same logistic step into one high-level step.

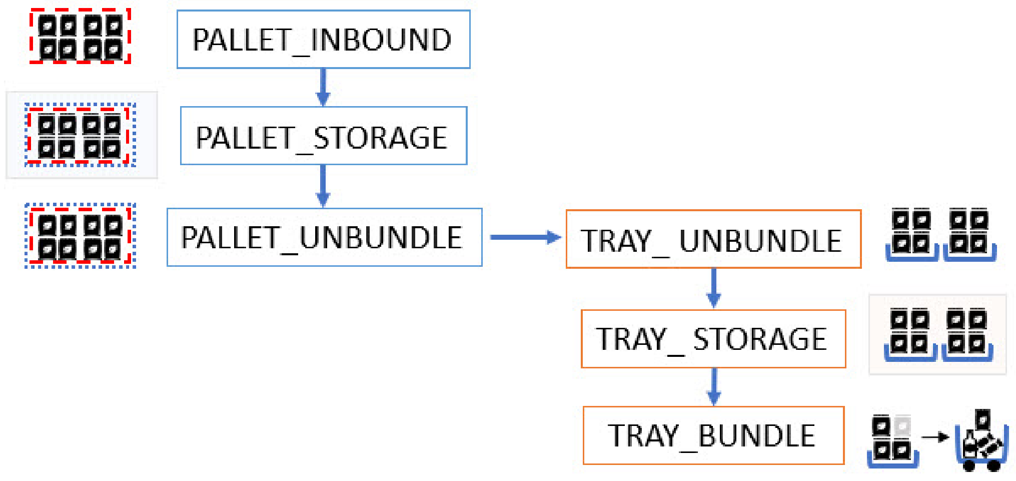


Figure 3.5: Example of High Level process in Warehouse Automation System

Low-level concept can be explained by an example illustrated in Figure 3.6. Four locations in the warehouse are passed by a pallet, generating four different activities. First, the pallet goes to receiving station number 2 at inbound area. Next, it moves to location1 and location2 and finally transfers to conveyor belt number 5. These four low-level steps are aggregated into 1 high-level step under PALLET_INBOUND.

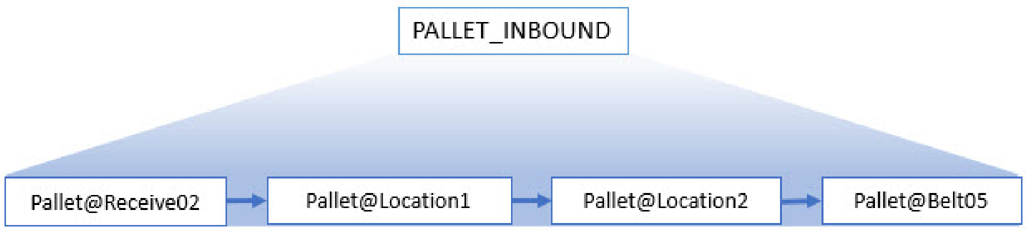


Figure 3.6: Example of Low Level process in Warehouse Automation System

3.3 Data Context

In this section, we aim to understand the complexity of warehouse data including scope and requirement needed for artifact-centric process mining technique. This is related to data understanding step in CRISP-DM on Section 1.4.

3.3.1 Scope and Complexity

Machine's Communication Messages

In the system, five modules are working together to control each component of the system via machine's communication messages. One of the modules is a central module called a Conductor that orchestrates the other modules. Each module has a bidirectional communication with the Conductor to send and respond to communication messages. Meanwhile, other modules can communicate only with Conductor but not between each other as shown in Figure 3.7. Moreover, each module has different number of communication messages.

These messages are what were indeed recorded in the system and considered as raw input data in this study. This raw data will be later transformed to obtain sequences of events per item for process mining.

All messages contain Transportation Unit ID (TsuID) which is used to identify pallets and trays in the system.

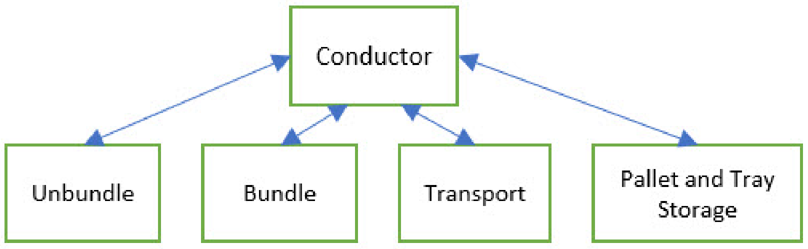


Figure 3.7: Communication Modules in Warehouse Automation System

Information of all modules in WAS are briefly described below.

1. Conductor: a control module that communicates with other modules in the system.
2. Unbundle: a module for "Unbundling".
3. Bundle: a module for "Bundling".
4. Transport: a module for "Transporting product from place to place" and controlling the flows of pallets and trays in the whole system.
5. Pallet and Tray Storage: a module for "Storing and Retrieving products in both Pallet and Tray Storage".

In total, there are in total 59 message tables that are used in the system combined from all modules.

Types of Messages

Messages can be classified into two types: first is related to system's order and second is related to system's action.

In this project, we consider only message tables that are related to system's action because message tables relevant to order have their own separated process flows. They interact with action-driven messages only when replenishment order or end customers' orders arrive. However, there are some arbitrary interaction between these two types of messages suggesting that some order-driven messages could interrupt action-driven messages anytime. One example is the case that a replenishment order arrives and is sent to action message to move a pallet out of the storage to Unbundle station. However, the order can be canceled by a certain reason. Hence, all events that occur must be rolled back meaning that the pallet needs to go back to the storage.

Moreover, including all messages for data analysis have been studied in a previous research by a TU/e Master student, K.W. (Koen) Verhaegh, as a result, all messages can generate a huge spaghetti model posing the difficulties to interpret the results [13]. Besides, this cannot perfectly represent actions in the main system due to the aforementioned cases that certain processes of order-driven events can execute the change in action-driven message at any time throughout the whole process.

To conclude, we only focus on system action messages and ignore order-driven messages in this study.

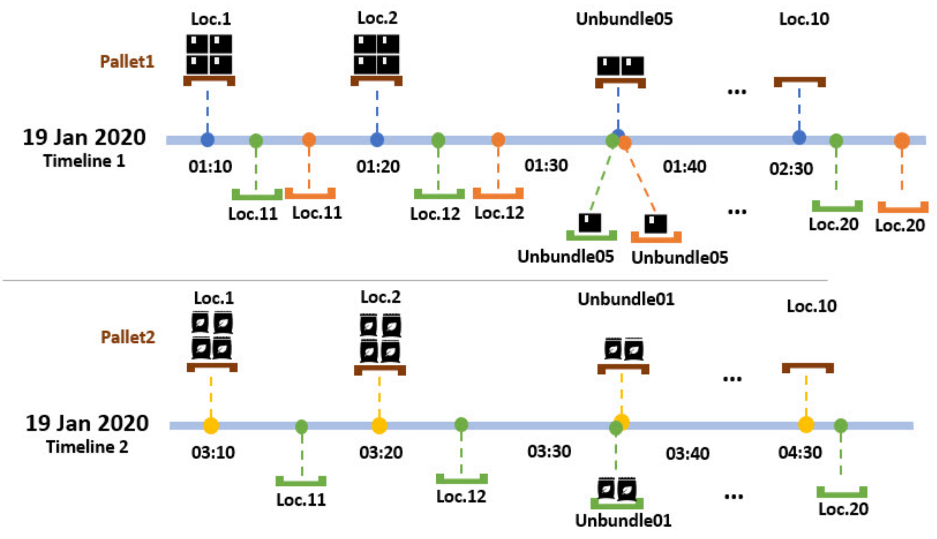
Complex Data Structure

Apart from the data flow in the system that hinders data understanding, data structure and the records in the data themselves are also complicated. Unlike a typical relational database that links relations between tables by primary and foreign keys, source data of WAS are communication messages without unique keys or identifiers for the tables, posing a challenge to understand the whole process from the data.

Figure 3.8 shows two simplified message tables: one from Transportation module and one from Unbundle module. The relations between these modules use TsuID as a linkage. In this example, there are two items – pallet and tray – with four items – two pallets and two trays. First, we focus on two pallets. The Pallet1 has multiple records in Transportation table and two records in Unbundle table. Meanwhile, Pallet2 has multiple records in Transportation table and one record in Unbundle table.

Considering timeline 1 shown at the top of Figure 3.8, the pallet activities, which are recorded in Transportation table, starts when Pallet1 containing boxes of water sequentially moves to location 1, 2 and ends at location 10. In the meantime, both Tray1 (green) and Tray2 (orange) are forwarded to location 11, 12 and end at location 20. In the message table, Pallet1 in Unbundle table contains two records for Tray1 and Tray2 at Unbundle station 05. Meanwhile, both trays also have a record of receiving products at Unbundle station 05 in this table as well. Afterward, both trays are stored and used until becoming empty allowing the start of a new round. However, we simplify the example by focusing only on Transportation and Unbundle tables. Hence, all records that do not belong to these two tables are not fully shown.

Next round is shown in timeline 2 as appeared in the middle of Figure 3.8. Pallet2 enters the system and moves to locations 1, 2 and 10, which are in the same route to Pallet1. Meanwhile, it goes to Unbundle station 01 and transfers snacks to Tray1. In this round, it can be seen that Tray1 has been reused by having the same TsuID from the previous cycle.



Records from Transportation Module

TsuID	Activity	Date and Time
Pallet1	MoveToLoc1	19-01-20 01:10
Pallet1	MoveToLoc2	19-01-20 01:20
Pallet1	⋮	⋮
Pallet1	MoveToLoc10	19-01-20 02:30
Tray1	MoveToLoc11	19-01-20 01:15
Tray1	MoveToLoc12	19-01-20 01:25
Tray1	⋮	⋮
Tray1	MoveToLoc20	19-01-20 02:35
Tray2	MoveToLoc11	19-01-20 01:16
Tray2	MoveToLoc12	19-01-20 01:26
Tray2	⋮	⋮
Tray2	MoveToLoc20	19-01-20 02:36
Pallet2	MoveToLoc1	19-01-20 03:10
Pallet2	MoveToLoc2	19-01-20 03:20
Pallet2	⋮	⋮
Pallet2	MoveToLoc10	19-01-20 04:30
Tray1	MoveToLoc11	19-01-20 03:15
Tray1	MoveToLoc12	19-01-20 03:25
Tray1	⋮	⋮
Tray1	MoveToLoc20	19-01-20 04:35

Message from Unbundle Module

TsuID	Activity	Date and Time	Product
Pallet1	Unbundle05	19-01-20 01:35	water
Pallet1	Unbundle05	19-01-20 01:36	water
Tray1	Unbundle05	19-01-20 01:35	water
Tray2	Unbundle05	19-01-20 03:36	water
Pallet2	Unbundle01	19-01-20 02:15	snack
Tray1	Unbundle01	19-01-20 02:15	snack

Figure 3.8: Example of data structure in Warehouse Automation System

Identical Events with Different Behaviors

In the source data, two identical events that are generated from different machine can have different behaviors. For instance, PalletInPosition can be from automatic Unbundle-station or manual Unbundle-station, where recording approaches are different. At an automatic station, products are distributed into trays automatically and the difference between each consecutive PalletInPosition represents the time spent in that distribution. In contrast, a manual station requires a working staff to manually press a button to move pallet into a correct position and the event is recorded every button pressing.

Different ways of event recording can lead to confusion, but it is decided not to discard any data of these to remain maximal information as possible with the awareness of the variation in recording.

3.3.2 Requirements for Artifact-centric Process Mining

This section describes the main requirements for artifact-centric process mining and explains how the source data are not satisfied for these requirements.

3.3.2.1 Unique identifier

From the example provided in Figure 3.8, there is no unique identifier. In WAS, both pallet and tray IDs could be reused for many times. Hence, defining lifecycle is necessary for differentiating pallets and trays that are used in different rounds. Hence, we could obtain unique identifier as a result.

3.3.2.2 Needs of Complete Lifecycle

Only pallets and trays that contain complete cycles are considered because they show the complete events that occur in the system and all rolling-back actions that create incomplete flows will be discarded from the analysis. In addition, analyzing end-to-end process requires both pallet and tray (with interrelation) to have fully complete cycles. Hence, this information facilitates the understandings of bottleneck or the findings of the causes of delay in the system.

3.3.2.3 Same Aspect Data

The utilized data are communication messages between machines. Hence, the names of event are recorded from the aspect of machine module. For example, an event being recorded in transportation module can be named as

"ExitPalletStorage". However, the exact meaning of this event is, "Pallet exits the transportation module and enters Pallet Storage". Naming approach can be confusing when considering other events from a variety of modules. It could be a case that event from Unbundle module is named "PalletExit", which actually means "Pallet exits the Unbundle station".

As can be seen from the above example, all event names in the modules need to be revised accordingly, and the aspects, from which the new names are generated, should be consistent.

3.3.2.4 Including Only Necessary Data for Analysis

In this study, we decide to discard records of empty trays. The main reason is that an empty tray has an arbitrary movement and it could go throughout the whole system, especially the routes between Tray Storage and Empty Tray Buffers. Moreover, it could be parked in any location in the storage. This is due to the management that WAS is configured to balance the flows and number of trays in the system. From discussion with domain expert, this information is not interesting for route and performance analysis and also generates confusion, it is thus excluded from this study.

3.3.2.5 Suitable level of Data Granularity

The current system specifies locations in Tray Storage by using many features including storage number, aisle, rack number, and positions in three dimensions. This approach of specification results in having in total 26,400 tray locations, excluding in-transit records within the storage. This huge number of tray locations generates difficulties in visualization and makes it hard to differentiate and understand interesting locations of TRAY_STORAGE HL step. It is thus necessary to simplify the information by discarding features that are less interesting: rack number and positions in three dimension. In practice, this is to lower the granularity of location recording, resulting in having 52 different locations in total, which is more suitable for analysis.

3.4 Problem Description

From the research question in section 1.2, this section describes the problem and how we can apply process mining techniques with the source data from WAS. Research question in section 1.2 has been defined at conceptual level and can be divided into several sub-research questions in more concrete details as follow.

RQ1 *Given a dataset of WAS, how could we understand the behavior of a warehouse and enable the extraction and integration of multiple items and their interrelations into a multi-dimensional event log that can represent multi-item dynamics?*

The complex and unstructured communication messages are difficult to entail the overall behavior, challenging the assessment of process efficiency.

Artifact-centric approach [2] as explained in Section 2.2.2 is here applied to our multi-item process data set from WAS. To apply this approach, there are prerequisites of items, sequences of events per items (routes) and time information per event for measuring system performance. However, the source data does not satisfy the requirement for artifact-centric process mining as explained in Section 3.3.2. To enable artifact-centric process mining approach in our data, more information is necessary, leading us to divide our research question RQ1 into 3 sub-questions.

- **RQ1.1** *Given a dataset of WAS, how could we define items, lifecycles and table relations stored in the data?*
- **RQ1.2** *Given a dataset of WAS, how could we obtain route for each item lifecycle including time information?*
- **RQ1.3** *Given a dataset of WAS, how could we obtain information of interrelation between items from the source data and information of interaction from domain knowledge?*

RQ2 *Given a set of WAS event logs, how could we apply the existing methods such as single-item route analysis, classification techniques and develop a meaningful notion?*

Established methods in process mining whose efficiency have been proven in a certain context (airport baggage handling system) shall be used with WAS.

RQ3 *Given a set of WAS event logs, how could we explore the behavior of multi-item process flow for a large warehouse and salient flows by textual description or visualization?*

These flows can elaborate the different behaviors between group of routes and the visualization can reveal possible underlying cause of deviation from optimum.

Chapter 4

Solution Proposal

In this chapter, we introduce the limitations and methods that are used to solve the research questions mentioned in Section 3.4.

We first mention about the limitations in this study on Section 4.4. Then, we explain our propose solution in the following sections. Figure 4.1 shows the overall steps of the methodology that are used in this project. To answer the RQ1, artifact-Centric approach, and multi-dimensional event log pre-processing is used to transform un-structured raw data to an event log explained in Section 4.1 and 4.2. These steps lead to the understanding of system behavior and identifying the relation between items. Next, the techniques for finding principle locations significantly representing different routes (PCA), reducing large number of principle locations to interesting locations (using K-means clustering), and grouping all routes into route clusters (multi-set), which are the existing techniques for single-item process elaborated in Section 4.3, are applied. The result from this step helps determine clusters of the routes in the system that have similar behavior which could answer RQ2. Then, we use the result from the route and performance analysis to develop two types of visualizations illustrating the overall routes with their clusters and the performance of the whole systems which could lead to the answer of RQ3.

4.1 Artifact-centric Process Analysis with Multi-items

To comprehend multi-item behavior from the raw data, it is necessary to determine items, lifecycles of items, relations and interaction between lifecycles.

Artifact-centric approach is suitable for multi-item process with many

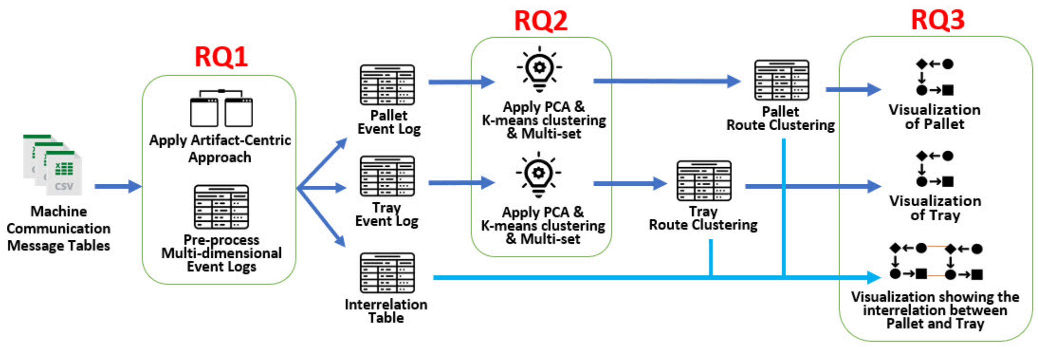

RQ1
RQ2
RQ3
Machine Communication Message Tables
Apply Artifact-Centric Approach
Pre-process Multi-dimensional Event Logs
Pallet Event Log
Tray Event Log
Interrelation Table
Apply PCA & K-means clustering & Multi-set
Apply PCA & K-means clustering & Multi-set
Pallet Route Clustering
Tray Route Clustering
Visualization of Pallet
Visualization of Tray
Visualization showing the interrelation between Pallet and Tray

Figure 4.1: The summary of data pipeline

items that have their own lifecycles involved. In addition, each item has its own identifier and interacts with other items belonging to different item class; the dynamic communication and intensive updates are prominent characteristics of the process.

Besides, un-structured raw data is retrieved from various modules in the system and there is no unique identifier that could well represent specific item consistently throughout the whole process. We could solve this problem by adopting the concept of artifact-centric process analysis with the four following steps shown in Figure 4.2. The procedure will be explained thoroughly in Chapter 5.

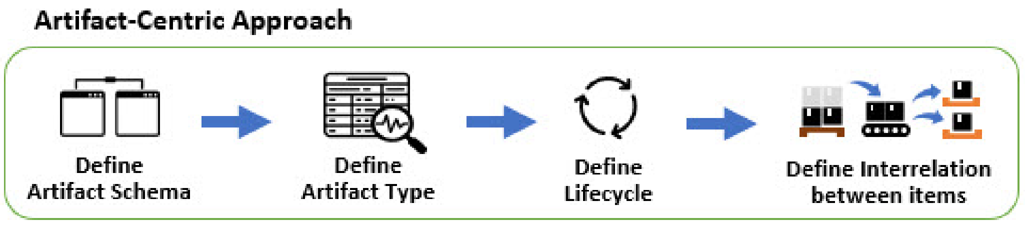


Figure 4.2: Overview of Artifact-centric Approach

Define Artifact Schema

As data is not in the form ready for analysis, each table has mixed identifiers in a column which could be either a pallet or a tray in our use case. Moreover, tables are created from different modules with different purposes; multiple tables can refer to the same item but record activity independently. In brief, the data of each item are diffused to many tables, each of which records different behavior of the item but still in chronological order.

This leads to the first step of artifact-centric approach: Artifact Schema Definition. In this step, it is necessary to determine appropriate tables that could represent all of the main system activities. Defining the schema needs cooperation with a domain expert to gain understanding on data.

Define Artifact Type

Once schema to be used is known, it is necessary to discover artifact type. The artifact type concept is applied for finding the relation across tables. To define artifact type, we start with selecting items of interest: pallet and tray in this study. We define identifier that could be used to link tables. In other words, the identifier is used as a table linkage. Then, the conditions (to be applied when retrieving data from tables) are established in order to extract only data of interest. We decide which data is essential for the analysis (to keep or to remove data) by using the conditions defined at this step.

Define Lifecycle

After artifact schema and artifact types are established, we need a concrete definition of lifecycle to reveal event-level interaction between items. While pallets and trays can be reused for many times, the current identifiers of them do not change by the repetitive uses. To distinguish the usages in different lifecycles, it is thus necessary to define new identifiers.

To do so, we start from studying the process model of each item at conceptual level. Then, we define the start activity and the end activity of pallet cycle and tray cycle. Furthermore, tray has a special behavior that it can contain several sub-cycles within its cycle because products on the tray could be stacked to a roll cage for many times until tray becomes empty, while all products on each pallet are always used at Unbundle station only once. This step also requires a domain expert to share knowledge on the system behavior and technical documents explaining the procedures of each module in the system. Note this step extracts the lifecycle conceptual level. We could not determine the actual event in the data yet.

Define Interrelation between Items

We now have artifact schema, artifact types and lifecycles. Then, behavioral dependencies between the artifact life cycles is need to be explored. This step could be done by reading technical documents and discussing with a domain expert to find the information to be used for finding the dependencies and to be able to understand end-to-end process.

With artifact-centric concept, we could answer sub-research question RQ1.1, which is relevant to the definition of items, lifecycles and table relations.

RQ1.1 *Given a dataset of WAS, how could we define items, lifecycles and table relations stored in the data?*

After obtaining the result from artifact-centric method, we can now relate each event to one item and to one specific lifecycle of the item. The newly identified lifecycle identifier can distinguish two different lifecycle executions of the same physical item. We can also relate all tables and have data that contains only important information that represents main activities in the system for the analysis.

This output will be used for creating a multi-dimensional event log in the following section.

4.2 Pre-processing Multi-dimensional Event Logs

Warehouse raw data are in form of mixed tables of communication message, which are not in the form ready for process mining as it requires identifiers, activities, and timestamps.

To convert the raw data into a usable event log, it is crucial to transform the data into an appropriate format that is easy for analysis.

Figure 4.3 illustrates the overview of data pre-processing. We have machine communication message tables as inputs. After applying four data pre-processing steps (define case and unique item identifier, define lifecycles and interrelations, and log extraction specification), we yield event log that has unique identifier per lifecycle as a result.

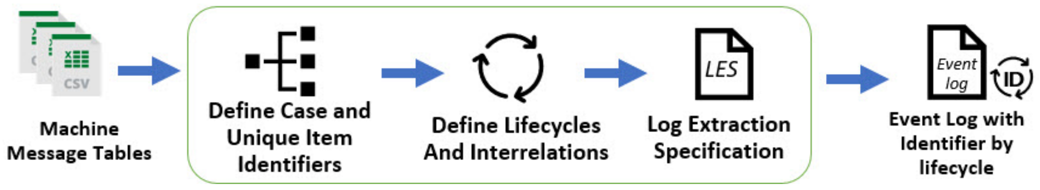

CSV
Machine Message Tables
Define Case and Unique Item Identifiers
Define Lifecycles And Interrelations
LES
Log Extraction Specification
Event log
ID
Event Log with Identifier by lifecycle

Figure 4.3: Overview of Multi-dimensional Event Log Pre-processing

Additionally, activities in the system are recorded from machine module aspect and do not satisfy the requirement in Section 3.3.2.3. It is difficult for an analyst or a domain expert to interpret because the same activity in different module could have different meaning. In this step, name correction is required to fix the meaning of each activity in the system to avoid confusion during the analysis.

To correct the records that explains the occurring activities in single aspect, a domain expert incorporates in defining and providing more information regarding the actual activities occurring in each machine module. As a result, we could rename the activity names to meaningful and logical keywords for all items.

We execute all pre-processing steps by creating several Scala scripts. We firstly search for each item type (pallet and tray) for all event-record activities of that item type, then rename all activities name based on one aspect, and sort all event records by item and time. Next, we identify item lifecycle identifier by using the lifecycle concept gained in Section 4.1 to acquire route for each item lifecycle with time information. Hence, we could answer sub-research question RQ1.2.

RQ1.2 *Given a dataset of WAS, how could we obtain route for each item lifecycle including time information?*

Afterward, we create interrelations between different items – pallet and tray – by extracting their relations from the data. Hence, we can now answer sub-research question RQ1.3, which is relevant to interrelation between items. **RQ1.3** *Given a dataset of WAS, how could we obtain interrelation information of interrelation between items from the source data and information of interaction from domain knowledge?*

The output from Section 4.1 and 4.2 will be used in the Section 4.3. Details about data pre-processing is given in Chapter 5.

4.3 Integrating Multi-dimensional Event Log with existing single-item techniques and visualizations

To validate the applicability of single-item techniques to multi-item process for route and performance analysis, the techniques and visualization previously proven to be successful in airport-baggage context are adopted.

4.3.1 Single-item techniques

In this study, we apply three techniques for route and performance analysis.

We use principle component analysis (PCA) as a first techniques. Principle components (PC) are calculated by sub-traces per HL step, rather than computing with the whole end-to-end traces. Considering sub-traces by HL step offers an understanding of significant behavior representing the part of the system.

PCA is used to reduce the dimension of routes in the warehouse system. The number of the remaining PCs is fixed to 3.

Besides, having three dimensions also allows visualizing the obtained result from PCA and using visualization to find an appropriate number of clusters in the next step.

Second, we use K-means clustering to uncover principle locations. We vary the value K per HL step because each HL step contains different number of locations. Using a fixed K for all HL steps are not appropriate, possibly leading to over-estimation or under-estimation of the number of clusters and failing to reflect the actual route clusters to be obtained at the end.

After applying K-means clustering, we find the centroid of each cluster and define the location closest to the centroid to be the location of interest. In order to determine an appropriate K value, we plot PCA-transformed data and visually estimate groups of points. Besides, we calculate the average Euclidean distance to corresponding centroid of each point and plot the average distance by the variation of K; we then select the "elbow" of curve as the appropriate K value, expected to be consistent with the visual estimation from PCA visualization. Then, interesting locations are selected for route clustering in the next step.

Third, multi-set abstraction is used after obtaining interesting locations by HL steps. We assume that those routes that transit the locations of interest for the same number of times might have similar behavior, for example, containing loop, having repeated steps, and tending to be grouped in the same cluster by K-means.

This step could lead to the answer of RQ2, which is relevant to the application of single-item methods to analyze and classify routes and make meaningful inference.

RQ2 *Given set of WAS event logs, how could we apply the existing methods such as single-item route analysis, classification techniques and develop a meaningful notion?*

We use the result yielded from this step in the subsequent visualizations. In-detailed single-item techniques used in this study will be elaborated in Chapter 6.

4.3.2 Visualization

There are two visualizations used in this study. The first one is the existing visualization used with airport system in Vanderlande (referred to as *single-item visualization*). The other one is the extended version showing interrelation between items, referred to as *multi-item visualization.*

We use the event log together with cluster information obtained from previous steps to generate both visualizations. As we define pallet and tray in the event log by using artifact-centric approach and determine clusters of routes by applying the existing single-item techniques, we can create the visualization for pallet and tray separately based on their HL variants in *single-item visualization* and merge both pallet and tray HL variants with their interrelation in *multi-item visualization.*

Single-item visualization

To create the *single-item visualization*, pallet and tray must have complete lifecycle because we would like to show average time spent in each HL step. The average spent time is color-coded in the visualization to represent system performance through cycle.

Each pallet/tray has its own visualization that contains all routes summarized from HL variants without considering the relation between items. In other words, we define HL variants on this visualization by pallet and tray separately under the conditions that pallet has complete cycle and tray has complete cycle but both pallet and tray that have interrelation do not need to be completed at the same time.

The generated visualization illustrates the performance and route clustering by HL variant in the system.

Multi-item visualization

This visualization is extended from *single-item visualization* by combining HL variants of items with interrelations. The outputted interaction between multiple items can be verified by this visualization, providing more insights beyond the dynamics of single item.

We pair up the pallet and tray with complete lifecycle that have interrelation between each other, as we adopt the concept in *single-item visualization* and extend it by combining HL variants. In our case, the paired pallet and tray are merged to create a new end-to-end HL variant, named pallet-tray HL variant.

We first create a table that lists all pallet-tray pairs in the system, named interrelation table. After that, we check all of the pallets' HL variants and trays' HL variants separately. Then, interrelation table is used to generate all possible combination of pallet's HL variant and tray's HL variant. The combination becomes our pallet-tray HL variant.

Afterward, we visualize end-to-end routes by using new HL variant to connect pallet and tray together. Hence, we could see the relation between pallet and tray under same newly-created HL variant.

From this step, RQ3, which is relevant to flow behavior, could be answered by these two visualizations.

RQ3 *Given a set of WAS event logs, how could we explore the behavior of multi-item process flow for a large warehouse and salient flows by textual description or visualization?*

The behaviors, notable from the visualization, are further investigated to identify possible underlying causes which will then be explained in text or by illustration. More information about the 2 visualizations is detailed in Chapter 7.

4.4 Limitations

This study focuses on a WAS of a specific Dutch supermarket company, which is a Vanderlande's client, where the generalizability to other WAS is limited. Specifically, suitable data structures might vary across systems. In addition, the data duration is restricted to one week, which might not capture some interesting information that require longer time span to analyze.

The locations mentioned in this thesis refer to the area of location, which would be adequate for analysis as confirmed by a domain expert. They are not indicating to the exact physical locations in the warehouse but are still referred to as "locations" hereafter in this thesis.

The clarity of result presentation in this thesis might be limited due to the impossibility to to disclose actual layout of the warehouse, under the agreement with the company. Results are described by text.

It is also important to realize that the warehouse we are considering has many Bundling and Unbundling stations. Routes to different Bundling or Unbundling stations are parallel routes that have the same number of locations, performing the same sequence of activities.

Chapter 5

Multi-dimensional Event Logs Pre-Processing

In the Chapters 3 and 4 , we gain comprehension at the conceptual level on WAS and the approach to handle data.

In this chapter, we explain, in details, on how to pre-process and transform source data to appropriate event log that we can use for subsequent analysis. The approach described here is in data preparation step, the third step of CRISP-DM mentioned in Section 1.4.

Data structure in WAS is different from a normal relational database that has a unique identifier used as primary key and foreign key linking tables in the database. This is because the interesting relations are all records in different tables referring to the same lifecycle of the same physical item. Moreover, that physical item has a unique identifier, but it can participate in multiple lifecycles after each other.

Because of the complicated way of storing data in tables, it is necessary to define log extraction specification (LES) by applying artifact-centric approach to enable the adoption of techniques for route and performance analysis.

LES is a technique to define the way to find two main information from a set of tables.

1. Case Notions: find the main table carrying the primary keys, each primary key value becomes a case identifier.
2. Event Types: define which timestamp attribute in which table defines which activity.

In general, LES could be used with both single-item and multi-item processes to extract all events of all types as structured records. Events that have the same case are grouped into a trace.

To apply LES with multi-item process, we use artifact-centric approach to construct LES that has multiple case notions by firstly defining items and defining LES for each item with its own identifier. Artifact-centric approach is required because of the complexity of stored data. There are two main steps to construct LES.

1. Find all tables that contain event records for a particular item. However, the collected tables may contain more event records that do not belong to the item for which we want to extract the log.

2. Define the LES on the selected tables and add selection predicates that only take those records for both case notion and event types that really belong to the item.

Figure 5.1 depicts the overview of pre-processing steps to create multi-dimensional event log in this study. The input is source data and knowledge from Chapter 3, which is then pre-processed using three main steps as described below, eventually yielding multi-dimensional event log. It should be noted that the first two steps could be done by using artifact-centric approach.

1. Defining case identifiers and unique item identifiers.

2. Defining lifecycle of each item and the interrelation between items.

3. Log Extraction Specification (LES).

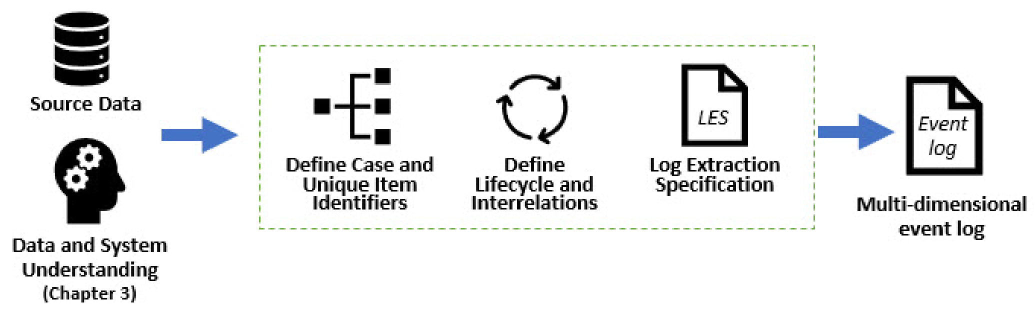

Source Data
Data and System Understanding (Chapter 3)
Define Case and Unique Item Identifiers
Define Lifecycle and Interrelations
LES
Log Extraction Specification
Event log
Multi-dimensional event log

Figure 5.1: Multi-dimensional Event Logs Pre-processing Overview

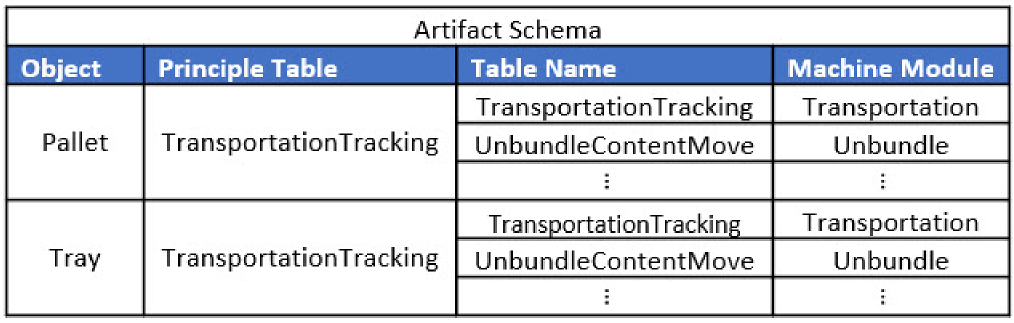
The artifact-centric approach itself encompasses four pre-processing steps, which are elaborated here. In Section 5.1, we first explain artifact-schema used for identifying the tables that should be considered together to acquire all relevant data for one item. Then, artifact type is used to specify conditions under which events are linked/related to an item in Section 5.2. We then create a new unique identifier that distinguishes different lifecycle, which is explained in Section 5.3, and we create interrelation table as explained in Section 5.4. Lastly, the final event logs from the source data is explained in Section 5.5.

5.1 Defining Artifact Schema

It is essential to find a principle table per item that has an identifier (primary key as TsuIDs) capable of correlating that item to other message tables. The principle table will be linked and integrated with other message tables to create a unified multi-dimensional event log that represents its multi-item dynamics covering the whole process.

In this study, a message from transportation module is chosen because the data inside the message table appear from start to end through the whole process. Then, we identify other related message tables that represent the main activities in the system. To do so, we review technical documents to gain comprehension on how the system records data together with discussing with domain expert getting to know interesting activities/tables which should be included.

Table 5.1 illustrates artifact schema in WAS. With the data we have, both pallet and tray have the same principle table called "Transportation-Tracking". The other are the tables that contain all main system activities. However, in this example, we focus on two tables for the sake of simplicity: TransportationTracking and UnbundleContentMove.



Artifact Schema			
Object	**Principle Table**	**Table Name**	**Machine Module**
Pallet	TransportationTracking	TransportationTracking	Transportation
		UnbundleContentMove	Unbundle
		⋮	⋮
Tray	TransportationTracking	TransportationTracking	Transportation
		UnbundleContentMove	Unbundle
		⋮	⋮

Table 5.1: Example of Artifact Schema

5.2 Defining Artifact Type

The objective of this step is to find related events that link to an item because the table contains more data records than just for the focused item. Hence, conditions is required to keep only the related records and the associated records from other tables linking to the main table in a specific way.

After we create artifact schema which is defined for each item from all the tables that hold information of this item and designates a main table holding a primary key identifier for the various cases.

Next, artifact type comes to define all records in the data per item. There are several components of artifact type as explained below:

1. Conditions when records of the main table actually define an item of this type (e.g.the main table record actually refers to a pallet/tray).
2. A list of event type, where each of event type defines following information:
    (a) The name of the activity, which requires domain-knowledge to map technical message names to human-understandable process steps.
    (b) The table and attribute from which the unique item identifier is taken for the event type. In other words, all events with same item identifier belong to the same item.
    (c) The table and attribute from which the timestamp information is taken.
    (d) A condition specifying whether a record in a specific table may be considered at all. This is to only include records actually related to the item type because a table can hold records of many different item types.

Having such LES then allows generating a query for each event type that extracts event records of 3 columns (name, identifier, and timestamp) for all events in the data. The output of running the queries for all event types is a uniform table of events having name, identifier, and timestamp as columns.

We need to define such LES, then we will obtain the needed event table.

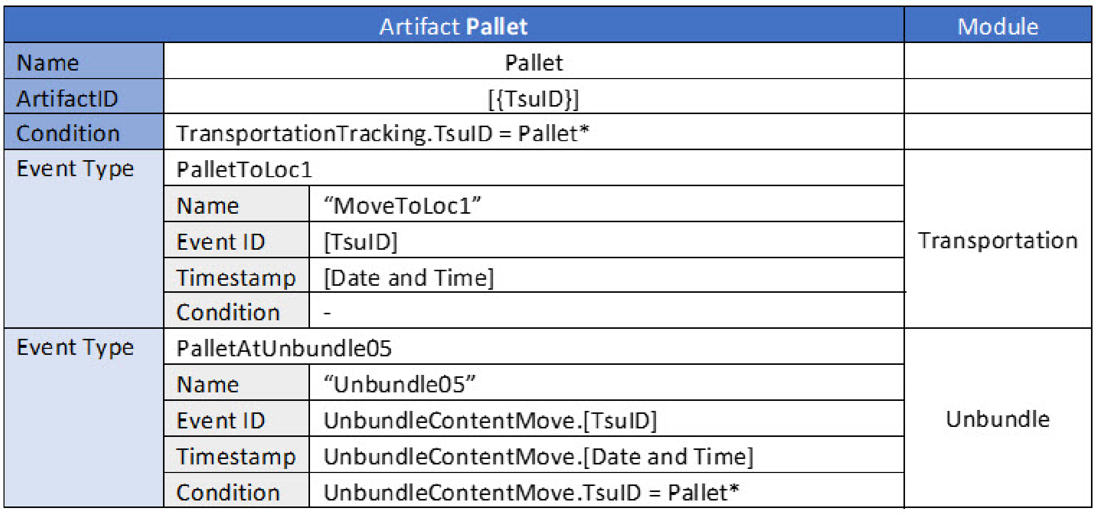
In this study, we create two artifact types; pallet and tray, which are shown in Table 5.2 and 5.3 respectively.

For example, we use our Artifact Schema in Table 5.1 to build pallet and tray artifact types. Hence, both pallet and tray have Transportation-Tracking table as a principle table. Another table used as an example is UnbundleContentMove table for both pallet and tray.

First, we consider pallet artifact type in Table 5.2. The primary key artifact ID is redefined by the value in TsuID column on principle table, TransportationTracking, starting with "Pallet". As a result, we could select only pallet records by using data format on TsuID column. Next, we define event types. Each event type could have different name, identifier, timestamp and conditions. This example show two event type; PalletToLoc1 and PalletAtUnbundle05 coming from different tables. Considering event type PalletToLoc1, the event is from TransportationTracking table, and consequently event ID is redefined by TsuID. No event type condition is applied. Meanwhile, event type PalletAtUnbundle05 comes from another table, UnbundleContentMove, having event ID from TsuID column, and event type condition pertaining only records having TsuID starting with "Pallet".

Similarly, the same concept is applied to tray artifact type. We have TransportationTracking as principle table but it is necessary to specify the condition of having TsuID that starts with "Tray". First event type, TrayToLoc11, is the same as pallet's first event type but refers to a different location. Furthermore, event type TrayAtUnbundle05 is from UnbundleContentMove table having TsuID as event ID, necessitating the establishment of event type condition to include only TsuIDs starting with "Tray".

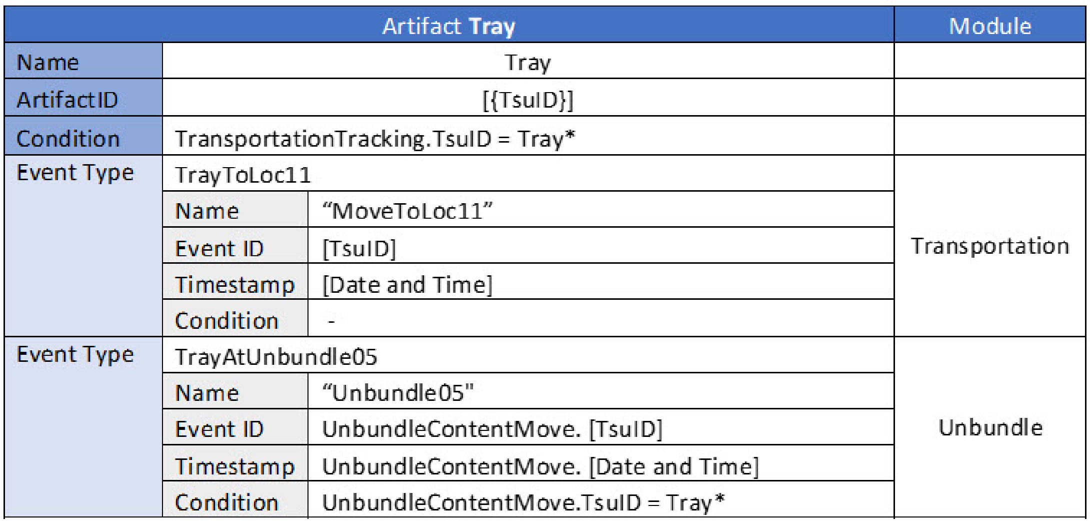
For Timestamp attribute, the data in all message tables have column *Date and Time*. Therefore, this column is used to redefine Timestamp in all selected tables for creating artifact type.



Artifact **Pallet**			Module
Name		Pallet	
ArtifactID		[{TsuID}]	
Condition		TransportationTracking.TsuID = Pallet*	
Event Type	PalletToLoc1		Transportation
	Name	“MoveToLoc1”	
	Event ID	[TsuID]	
	Timestamp	[Date and Time]	
	Condition	-	
Event Type	PalletAtUnbundle05		Unbundle
	Name	“Unbundle05”	
	Event ID	UnbundleContentMove.[TsuID]	
	Timestamp	UnbundleContentMove.[Date and Time]	
	Condition	UnbundleContentMove.TsuID = Pallet*	

Table 5.2: Example of Pallet artifact Type

After obtaining artifact type consisting of all related tables from artifact Schema, we can now create one unified table that has all items and events



Artifact Tray			Module
Name	Tray		
ArtifactID	[{TsuID}]		
Condition	TransportationTracking.TsuID = Tray*		
Event Type	TrayToLoc11		Transportation
	Name	“MoveToLoc11”	
	Event ID	[TsuID]	
	Timestamp	[Date and Time]	
	Condition	-	
Event Type	TrayAtUnbundle05		Unbundle
	Name	“Unbundle05"	
	Event ID	UnbundleContentMove. [TsuID]	
	Timestamp	UnbundleContentMove. [Date and Time]	
	Condition	UnbundleContentMove.TsuID = Tray*	

Table 5.3: Example of Tray Artifact Type

for each pallet and tray. All items in this table are grouped by TsuID and ordered by Date and Time. In addition, the activity names recorded in the data may be unclear or duplicated between pallet and tray. Therefore, we could redefine to meaningful and logical name at this step. We call the tables obtained from this step as an intermediate event log. Next, we need to create new TsuID that is unique across different lifecycles.

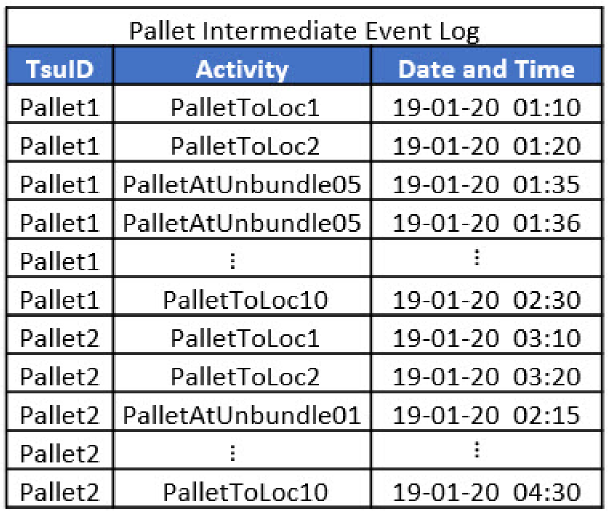
Table 5.4 and 5.5 are examples of intermediate event logs gained from this step with new meaningful name. These examples are created and in line with the timelines in Figure 3.8 in Section 3.3.1 which will also be used in the following section.

5.3 Creating new unique identifier by lifecycle

In this step, we apply artifact lifecycle together with the LES from Section 5.2 called as intermediate event logs of pallet and tray in Table 5.4 and 5.5 respectively. Artifact lifecycle is involved in this step because the intermediate event logs contain multiple cycles that are not distinguished. Hence, we need to re-define identifier that groups events by lifecycle, not by TsuID.

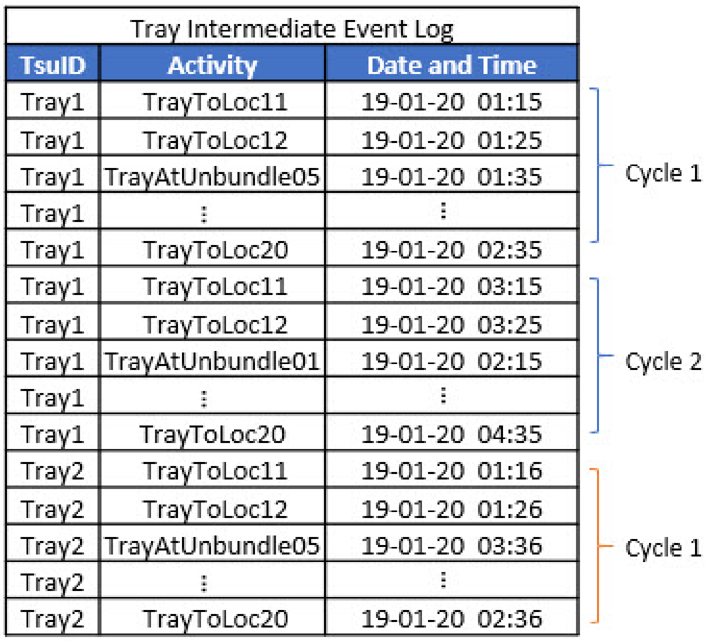
We first explain the general idea how we could create a new unique identifier by lifecycle.

1. Define start and end point of a lifecycle by using domain knowledge.



Pallet Intermediate Event Log		
TsuID	Activity	Date and Time
Pallet1	PalletToLoc1	19-01-20 01:10
Pallet1	PalletToLoc2	19-01-20 01:20
Pallet1	PalletAtUnbundle05	19-01-20 01:35
Pallet1	PalletAtUnbundle05	19-01-20 01:36
Pallet1	⋮	⋮
Pallet1	PalletToLoc10	19-01-20 02:30
Pallet2	PalletToLoc1	19-01-20 03:10
Pallet2	PalletToLoc2	19-01-20 03:20
Pallet2	PalletAtUnbundle01	19-01-20 02:15
Pallet2	⋮	⋮
Pallet2	PalletToLoc10	19-01-20 04:30

Table 5.4: Example of Pallet Intermediate Event Log



Tray Intermediate Event Log			
TsuID	Activity	Date and Time	
Tray1	TrayToLoc11	19-01-20 01:15	Cycle 1
Tray1	TrayToLoc12	19-01-20 01:25	
Tray1	TrayAtUnbundle05	19-01-20 01:35	
Tray1	⋮	⋮	
Tray1	TrayToLoc20	19-01-20 02:35	
Tray1	TrayToLoc11	19-01-20 03:15	Cycle 2
Tray1	TrayToLoc12	19-01-20 03:25	
Tray1	TrayAtUnbundle01	19-01-20 02:15	
Tray1	⋮	⋮	
Tray1	TrayToLoc20	19-01-20 04:35	
Tray2	TrayToLoc11	19-01-20 01:16	Cycle 1
Tray2	TrayToLoc12	19-01-20 01:26	
Tray2	TrayAtUnbundle05	19-01-20 03:36	
Tray2	⋮	⋮	
Tray2	TrayToLoc20	19-01-20 02:36	

Table 5.5: Example of Tray Intermediate Event Log

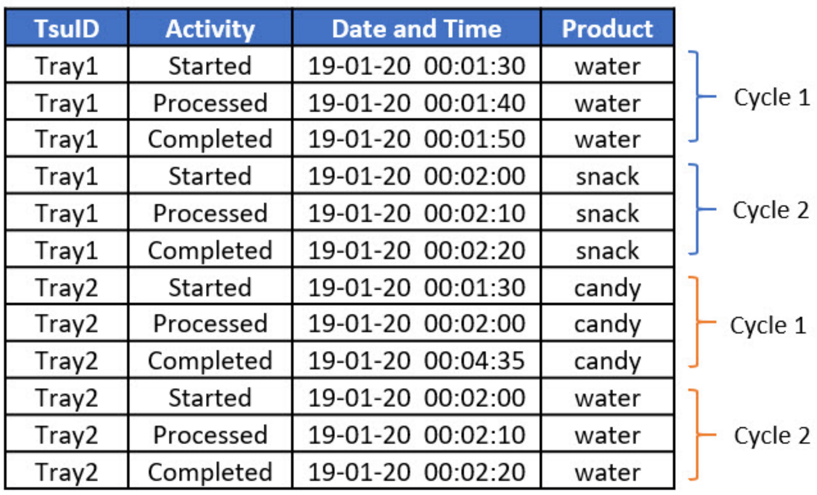
2. Extract the intervals of lifecycles into a temporary table. This step is to obtain start and end timestamps of each lifecycle.

3. Refine TsuID attribute by appending the number of the lifecycle to the TsuID for all events belonging to the same lifecycle.

At this point, the current intermediate event logs have TsuID as a case identifier specifying pallet or tray. However, using only TsuID is not sufficient for creating a unique case identifier because any pallet and tray can be reused but the corresponding TsuIDs remain unchanged. Consequently, lifecycle concept is required and used in defining the unique pallet and tray case identifiers that occur in different lifecycle.

A simplified example in Table 5.6 demonstrates the issue of duplicated case identifier. We select column "TsuID" as our identifier. There are two unique tray IDs; Tray1 and Tray2.

Given that start point is the time when activity becomes "Started" and that end point is the time when activity becomes "Completed", it can be seen that TsuID Tray1 or Tray2 has two cycles emerging in the system. With existing ID, we could not clearly identify items by lifecycle.



TsuID	Activity	Date and Time	Product	
Tray1	Started	19-01-20 00:01:30	water	
Tray1	Processed	19-01-20 00:01:40	water	Cycle 1
Tray1	Completed	19-01-20 00:01:50	water	
Tray1	Started	19-01-20 00:02:00	snack	
Tray1	Processed	19-01-20 00:02:10	snack	Cycle 2
Tray1	Completed	19-01-20 00:02:20	snack	
Tray2	Started	19-01-20 00:01:30	candy	
Tray2	Processed	19-01-20 00:02:00	candy	Cycle 1
Tray2	Completed	19-01-20 00:04:35	candy	
Tray2	Started	19-01-20 00:02:00	water	
Tray2	Processed	19-01-20 00:02:10	water	Cycle 2
Tray2	Completed	19-01-20 00:02:20	water	

Table 5.6: Simplified example of duplication ID in different lifecycles

After understanding the reason why the LES from 5.2 could not be used to create final event log, the procedure to create new unique identifier by lifecycle is applied.

Defining start and end point of a lifecycle

We need to split a sequence of events for the same TsuID based on domain information when a lifecycle starts or ends. Hence, we could obtain start and end point of pallet and tray lifecycle as follows.

- Pallet Lifecycle

  - Start: the time when pallet enters the system.
  - End: the time when pool and slave pallets are de-stacked.

- Tray Lifecycle
  - Start: the time when products on pallets are distributed into trays.
  - End: the time when a tray becomes empty.
    - Sub-cycle Start: the time when products on pallets are distributed into trays or after a tray has finished Bundling activity.
    - Sub-cycle End: the time when a tray has finished Bundling activity or a tray becomes empty.

Extracting the intervals of lifecycles into a temporary table

For each pallet and tray, we create a new table containing start and end timestamps of lifecycle. Hereafter, we refer to this new table as start-end table.

As mentioned in Section 5.2, our pallet and tray intermediate event logs are grouped by TsuID and ordered by Date and Time. Hence, we can use these logs directly. While we know start and end points of lifecycle, mapping start and end points with real records in the intermediate event logs is still required to find actual timestamp.

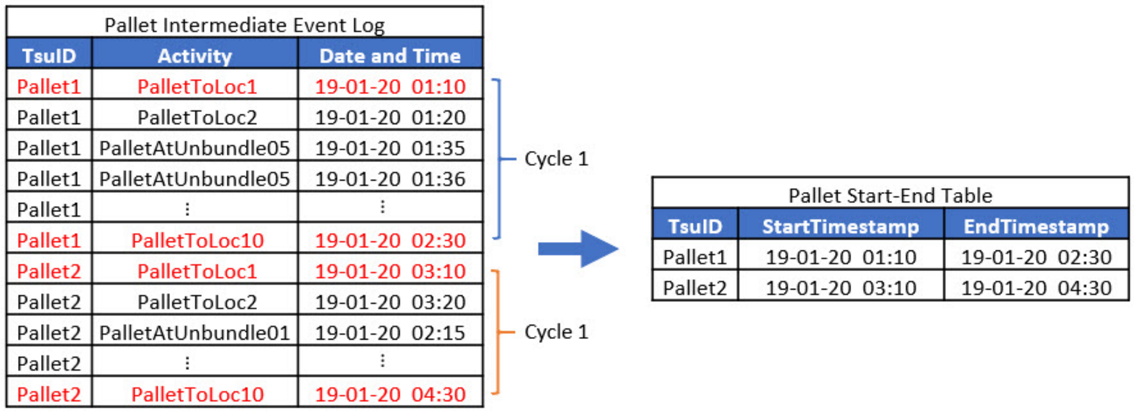
We use Table 5.4 and 5.5 as an example to show how we can find start and end timestamp by items.

Pallet lifecycle starts at the time point when pallet enters the system and ends at the time point when pool and slave pallets are de-stacked. Here, we suppose that activity PalletToLoc1 is the first activity recorded by the system and that PalletToLoc10 is the activity that pallet goes to the place where pool and slave are separated.

On the other hand, tray starts lifecycle when products on pallets are distributed into trays and ends when a tray becomes empty. Here, we suppose that activity starting with TrayAtUnbundle in the intermediate event log equivalent to the start point and that TrayToLoc20 refers to the end activity when tray status changes from tray with some product to an empty tray.

Once we finish mapping activity in the intermediate event log with start and end point from lifecycle, we can gain the actual start and end timestamps recorded with activities that are defined in the mapping step. Then, we can create start-end record. However, the records of some items in the intermediate event log, which have either start or end activity but do not both of them, are discarded because they could not represent complete cycle where the motivation is given in Section 3.3.1.

Table 5.7 and 5.8 are the output in this step. We could see that Table 5.7 consists of two records because both Pallet1 and Pallet2 have only one cycle. Meanwhile, Table 5.8 has three records. Because Tray1 is reused, it has two records referring to two cycles, while Tray2 has only one cycle.



Cycle 1
Cycle 1

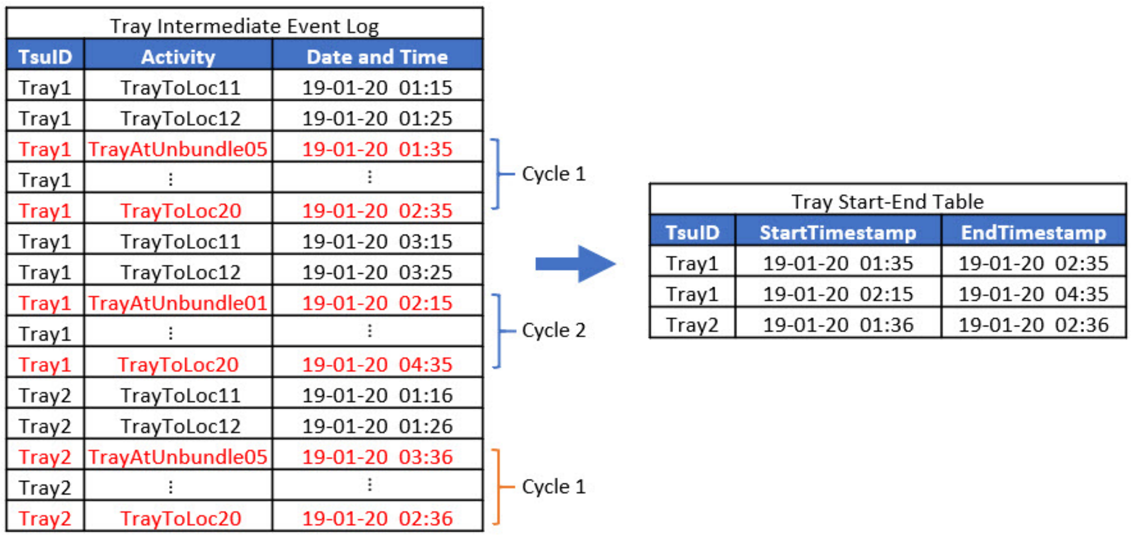
Pallet Intermediate Event Log

TsuID	Activity	Date and Time
Pallet1	PalletToLoc1	19-01-20 01:10
Pallet1	PalletToLoc2	19-01-20 01:20
Pallet1	PalletAtUnbundle05	19-01-20 01:35
Pallet1	PalletAtUnbundle05	19-01-20 01:36
Pallet1	⋮	⋮
Pallet1	PalletToLoc10	19-01-20 02:30
Pallet2	PalletToLoc1	19-01-20 03:10
Pallet2	PalletToLoc2	19-01-20 03:20
Pallet2	PalletAtUnbundle01	19-01-20 02:15
Pallet2	⋮	⋮
Pallet2	PalletToLoc10	19-01-20 04:30

Pallet Start-End Table

TsuID	StartTimestamp	EndTimestamp
Pallet1	19-01-20 01:10	19-01-20 02:30
Pallet2	19-01-20 03:10	19-01-20 04:30

Table 5.7: Example of Pallet Start-End Table



Cycle 1
Cycle 2
Cycle 1

Tray Intermediate Event Log

TsuID	Activity	Date and Time
Tray1	TrayToLoc11	19-01-20 01:15
Tray1	TrayToLoc12	19-01-20 01:25
Tray1	TrayAtUnbundle05	19-01-20 01:35
Tray1	⋮	⋮
Tray1	TrayToLoc20	19-01-20 02:35
Tray1	TrayToLoc11	19-01-20 03:15
Tray1	TrayToLoc12	19-01-20 03:25
Tray1	TrayAtUnbundle01	19-01-20 02:15
Tray1	⋮	⋮
Tray1	TrayToLoc20	19-01-20 04:35
Tray2	TrayToLoc11	19-01-20 01:16
Tray2	TrayToLoc12	19-01-20 01:26
Tray2	TrayAtUnbundle05	19-01-20 03:36
Tray2	⋮	⋮
Tray2	TrayToLoc20	19-01-20 02:36

Tray Start-End Table

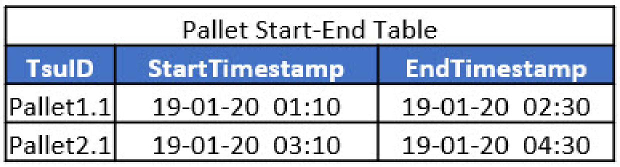
TsuID	StartTimestamp	EndTimestamp
Tray1	19-01-20 01:35	19-01-20 02:35
Tray1	19-01-20 02:15	19-01-20 04:35
Tray2	19-01-20 01:36	19-01-20 02:36

Table 5.8: Example of Tray Start-End Table

However, we still have the duplicated TsuID for Tray1 and the other pallets and trays can also be reused in the future. Therefore, we create a serial number of the cycle for each TsuID running in chronological order.

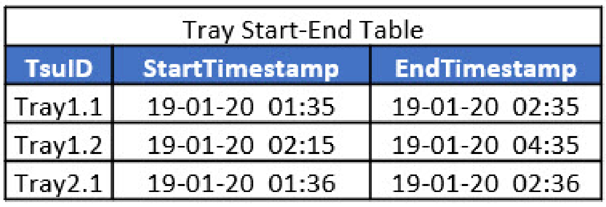
We append the cycle number at the end of existing identifier. With this approach, we could track how many times those items are reused and gain the knowledge on cycle order.

The TsuID in Table 5.7 and 5.8 is redefined as shown in Table 5.9 and 5.10.



Pallet Start-End Table		
TsuID	**StartTimestamp**	**EndTimestamp**
Pallet1.1	19-01-20 01:10	19-01-20 02:30
Pallet2.1	19-01-20 03:10	19-01-20 04:30

Table 5.9: Example of Pallet Start-End Table with new unique identifiers



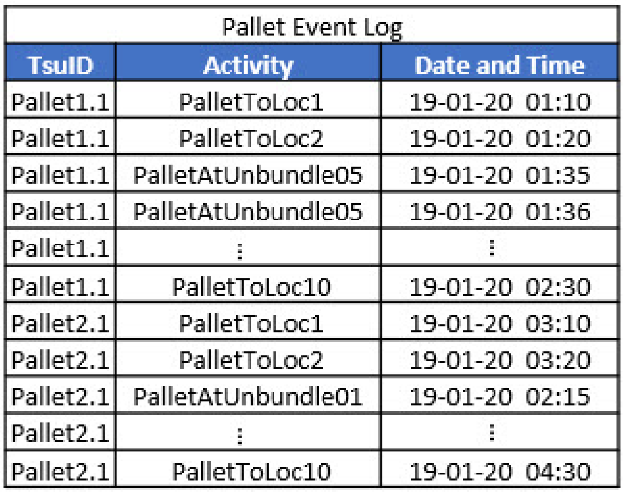
Tray Start-End Table		
TsuID	**StartTimestamp**	**EndTimestamp**
Tray1.1	19-01-20 01:35	19-01-20 02:35
Tray1.2	19-01-20 02:15	19-01-20 04:35
Tray2.1	19-01-20 01:36	19-01-20 02:36

Table 5.10: Example of Tray Start-End Table with new unique identifiers

Refining TsuID attribute

Lastly, the start-end table is used to assign new TsuID in our intermediate event log. By checking the timestamp of the records in the intermediate event log, we examine if the time falls between any start and end timestamp record in the start-end table. If the time falls into any range of start-end pair, a new TsuID is assigned to that record. Otherwise, the record is discarded because we consider that this record is not in any cycles in start-end table or not in domain expert's interests as explained in Section 3.3.1. The resultant event logs are yielded and shown in Table 5.11 and 5.12

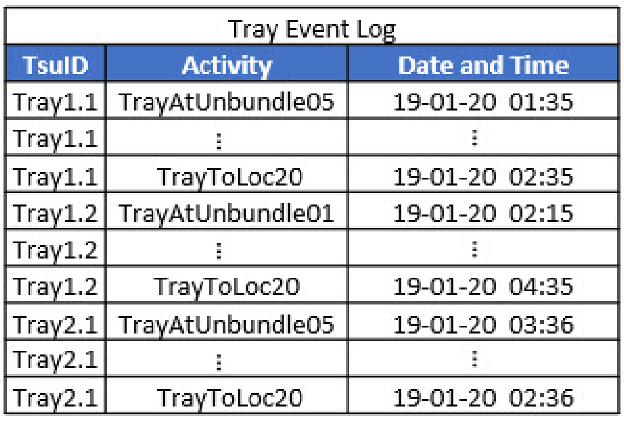
After finishing all three steps, we could gain the final event log that has new unique identifier by lifecycle, activity and timestamp columns which represent each physical item in the system as a result.



Pallet Event Log

TsuID	Activity	Date and Time
Pallet1.1	PalletToLoc1	19-01-20 01:10
Pallet1.1	PalletToLoc2	19-01-20 01:20
Pallet1.1	PalletAtUnbundle05	19-01-20 01:35
Pallet1.1	PalletAtUnbundle05	19-01-20 01:36
Pallet1.1	⋮	⋮
Pallet1.1	PalletToLoc10	19-01-20 02:30
Pallet2.1	PalletToLoc1	19-01-20 03:10
Pallet2.1	PalletToLoc2	19-01-20 03:20
Pallet2.1	PalletAtUnbundle01	19-01-20 02:15
Pallet2.1	⋮	⋮
Pallet2.1	PalletToLoc10	19-01-20 04:30

Table 5.11: Example of Pallet Event Log



Tray Event Log

TsuID	Activity	Date and Time
Tray1.1	TrayAtUnbundle05	19-01-20 01:35
Tray1.1	⋮	⋮
Tray1.1	TrayToLoc20	19-01-20 02:35
Tray1.2	TrayAtUnbundle01	19-01-20 02:15
Tray1.2	⋮	⋮
Tray1.2	TrayToLoc20	19-01-20 04:35
Tray2.1	TrayAtUnbundle05	19-01-20 03:36
Tray2.1	⋮	⋮
Tray2.1	TrayToLoc20	19-01-20 02:36

Table 5.12: Example of Tray Event Log

5.4 Creating Interrelation Table

Interrelation table allows identifying the relation between items and could shed more light on end-to-end process in warehouse.

We use start-end table obtained from Section 5.3 together with a message table containing interrelation information to create interrelation table.

First we explain how interrelations between items are recorded, then we explain how we transform the data to be in a more usable form.

5.4.1 Understanding existing interrelation data

There is one message table from Unbundle module containing interrelation information between pallet and tray. However, we could not straightforwardly use this table because data is not in a usable form. Therefore, we need to transform this table into a right format.

Table 5.13 is an example of the message table that has information about pallet and tray interrelation. In this example, there are five records in total, two for pallet and three for tray. Pallet identifier is stored in a column named SourceTsuID with its Unbundle station number 05 for Pallet1 and 01 for Pallet2 in column TargetTsuID. In contrast, tray identifier is recorded in TargetTsuID column and Unbundle information is stored in SourceTsuID. Both pallet and tray are linked by a column called OrderID. The OrderID is specific for one pallet with the distribution to many trays. It can be seen that the products in Pallet1 are distributed into Tray1, Tray2by the same OrderID 1 at Unbundle station 05, thus Pallet1 is related to Tray1 and Tray2 On the other hand, Products on Pallet2 are distributed to Tray1 at Unbundle station 01 by the OrderID 2. Consequently, Pallet2 has interrelation with Tray1. Apparently, the activities of pallet and tray are recorded independently and resulting records of them are not appearing in the same row.

5.4.2 Transformation to interrelation table

After we understand the existing interrelation in Section 5.4.1, data transformation is required to create interrelation table.

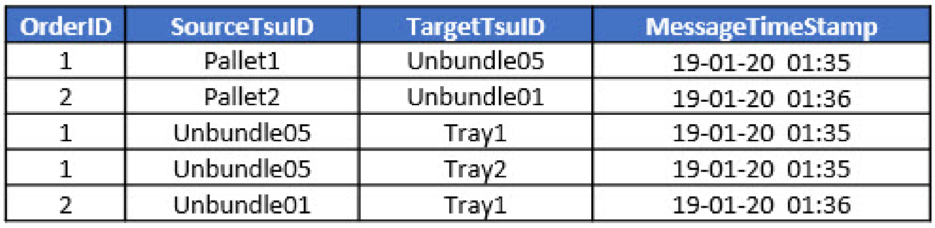
The first step is to replace the global TsuIDs with the refined TsuIDs. We use start-end tables – Table 5.9 and 5.10 obtained from Section 5.3 – to assign unique identifier. We do it by checking timestamp of each record if it falls between the start-end timestamp pairs in start-end table, identical to the step we assign new unique item identifier in Section 5.3. Hence, we could obtain unique identifier as shown in Table 5.14.

Afterward, we construct logic to create interrelation table as follows.

We split the interrelation table by classifying records as pallet and tray. We select OrderID and SourceTsuID columns that have SourceTsuID starting with “Pallet” as pallet records. Meantime, OrderID and TargetTsuID columns are used for tray records by having value starting with “Tray” in TargetTsuID. Both pallet and tray must have distinct records. Then, OrderID is utilized to connect pallet and tray which have interrelation together. The query used to extract interrelation from interrelation table is:

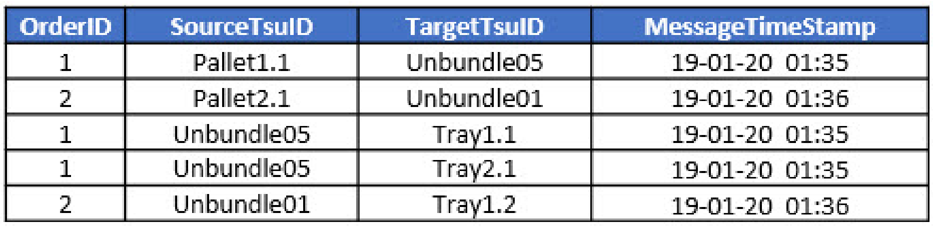
```
SELECT distinct (OrderID, SourceTsuId) as Pallet
FROM InterrationTable
WHERE SourceTsuId = Pallet∗

SELECT distinct (OrderID, TargetTsuId) as Tray
FROM InterrationTable
WHERE TargetTsuId = Tray∗

SELECT SourceTsuId, TargetTsuId
FROM Pallet JOIN Tray
WHERE Pallet.OrderId  = Tray.OrderId
```

As a result, we could now create interrelation table as shown in Table 5.15.



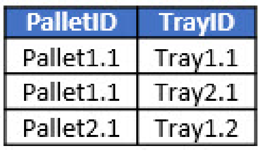
OrderID	SourceTsuID	TargetTsuID	MessageTimeStamp
1	Pallet1	Unbundle05	19-01-20 01:35
2	Pallet2	Unbundle01	19-01-20 01:36
1	Unbundle05	Tray1	19-01-20 01:35
1	Unbundle05	Tray2	19-01-20 01:35
2	Unbundle01	Tray1	19-01-20 01:36

Table 5.13: Example of message table for creating Interrelation Table



OrderID	SourceTsuID	TargetTsuID	MessageTimeStamp
1	Pallet1.1	Unbundle05	19-01-20 01:35
2	Pallet2.1	Unbundle01	19-01-20 01:36
1	Unbundle05	Tray1.1	19-01-20 01:35
1	Unbundle05	Tray2.1	19-01-20 01:35
2	Unbundle01	Tray1.2	19-01-20 01:36

Table 5.14: Example of message table with Unique Item Identifier for creating Interrelation Table



PalletID	TrayID
Pallet1.1	Tray1.1
Pallet1.1	Tray2.1
Pallet2.1	Tray1.2

Table 5.15: Example of Interrelation Table

5.5 Results of Log Extraction

After executing all event log pre-processing steps with one-week data in WAS, we achieve pallet event log consisting of 63,841 events and 1,585 cases, and tray event log consisting of 1,699,215 events and 55,189 cases. Each case regards to one lifecycle of an item; pallet and tray.

The pallet and tray event logs will be used with the existing single-item technique for route and performance analysis in the next chapter.

Chapter 6

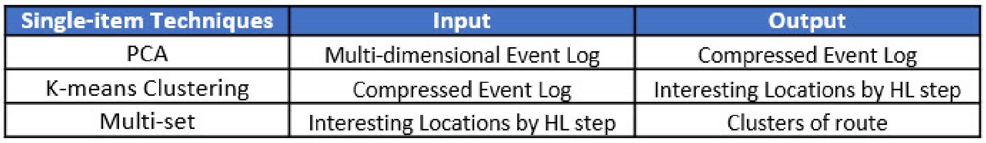
Route Summarization on Warehouse Items

After converting data into a single-item event log per item type as explained in Chapter 5, we then summarize route per item type. To do this, the existing single-item technique, as explained in Section 2.3, is applied. This chapter provides detailed information on how we configure parameters that are suitable for multi-dimensional event log from WAS and how we apply existing techniques with the data to cluster route and gain more understanding on WAS route behavior.

This chapter explains modeling steps of our methodology in CRISP-DM which are previously explained in Section 1.4.

A single-item technique comprising of three steps are applied in this study. PCA is applied with the objective to reduce data dimension and to extract only important features representing the whole data set. Once we have compressed data, we apply K-means clustering to obtain representative locations as interesting locations for route clustering in the subsequent step. Then, all interesting locations are applied with multi-set abstraction of interesting locations to create a cluster that aggregates all routes with similar behaviors into the same group.

Table 6.1 shows input and output of single-item techniques used in this study.



Single-item Techniques	Input	Output
PCA	Multi-dimensional Event Log	Compressed Event Log
K-means Clustering	Compressed Event Log	Interesting Locations by HL step
Multi-set	Interesting Locations by HL step	Clusters of route

Table 6.1: Input and Output of Single-item Techniques

6.1 PCA

In order to reduce dimensionality of data by removing redundant information, PCA is applied to sub-traces in each HL step, where each principle component (PC) captures characteristic sub-trace behavior. Only top three PCs capturing most data variants are kept.

6.1.1 Visited vs Unvisited Locations

In this study, locations in event log can be categorized into visited locations and unvisited locations. In 1-week warehouse data, there are some locations that are never visited by any pallet nor tray. While including and excluding unvisited locations when applying PCA may not significantly influence the learning of PC, they could impact results of the subsequent finding of interesting location and the clustering routes.

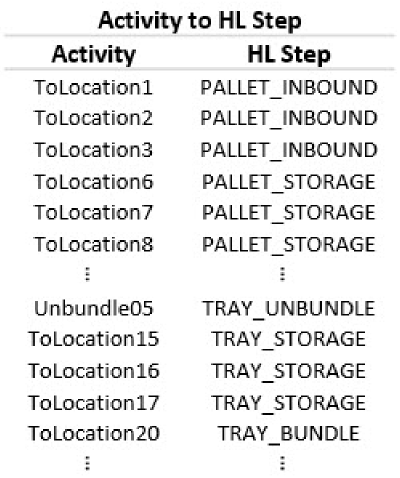
We do not consider unvisited locations once we pass the event log to PCA algorithm in this study because unvisited locations will have same PC information and they will be grouped together in the space of 3 PCs. Then, one of unvisited location will be selected as an interesting location. Afterward, unvisited location is used to create a cluster.

Nonetheless, the found locations of interest is subsequently utilized together with multi-set abstraction of interesting locations for creating cluster of routes. If unvisited location is considered as an interesting location, we will not be able to create any cluster from this location because there is no route passing this location. Hence, we yield zero cluster from this location that is supposed to be interesting. Besides, another location that should receive extra attention, which might generate cluster(s) that is more worthy for investigation, might not be picked up as the availability of interesting locations, resulting from K-means, has been partially allocated to the trivial unvisited location. This is undersirable and the chance of missing significant locations should be reduced by removing unvisited location from the analysis beforehand.

However, excluding unvisited locations cannot be done by straightforward removing the locations from the raw system log. It is rather done after generating the input matrix, which will be explained in the next step.

6.1.2 Preparing Data for PCA

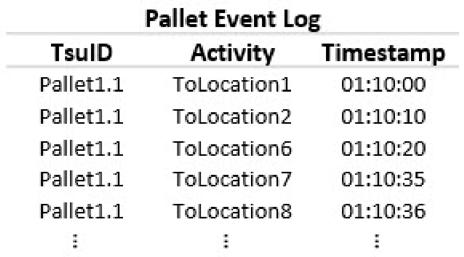
In generating input matrix for PCA, the first step requires an extra file that contains the information of mapping between activities and HL steps, named as mapping file. This file is manually created from our discussion with a domain expert; it is simply exemplified in Table 6.2. It is used together with pallet and tray event logs (with refined identifier) yield from Section 5.3 to append HL step information. Table 6.3 and 6.4 are examples of pallet and tray event logs.



Activity to HL Step

Activity	HL Step
ToLocation1	PALLET_INBOUND
ToLocation2	PALLET_INBOUND
ToLocation3	PALLET_INBOUND
ToLocation6	PALLET_STORAGE
ToLocation7	PALLET_STORAGE
ToLocation8	PALLET_STORAGE
⋮	⋮
Unbundle05	TRAY_UNBUNDLE
ToLocation15	TRAY_STORAGE
ToLocation16	TRAY_STORAGE
ToLocation17	TRAY_STORAGE
ToLocation20	TRAY_BUNDLE
⋮	⋮

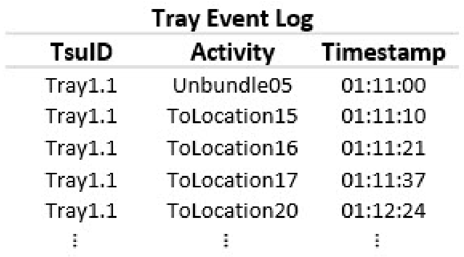
Table 6.2: Example of mapping file from Activities to HL step



Pallet Event Log

TsuID	Activity	Timestamp
Pallet1.1	ToLocation1	01:10:00
Pallet1.1	ToLocation2	01:10:10
Pallet1.1	ToLocation6	01:10:20
Pallet1.1	ToLocation7	01:10:35
Pallet1.1	ToLocation8	01:10:36
⋮	⋮	⋮

Table 6.3: Example of Pallet Event Log

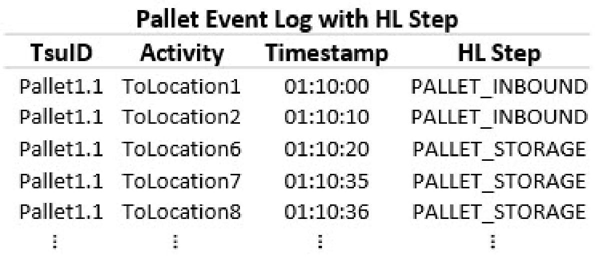


Tray Event Log

TsuID	Activity	Timestamp
Tray1.1	Unbundle05	01:11:00
Tray1.1	ToLocation15	01:11:10
Tray1.1	ToLocation16	01:11:21
Tray1.1	ToLocation17	01:11:37
Tray1.1	ToLocation20	01:12:24
⋮	⋮	⋮

Table 6.4: Example of Tray Event Log

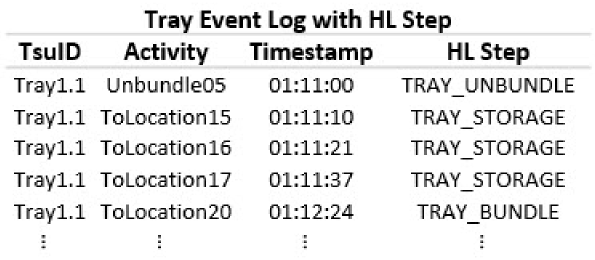
To append HL step information, we use Activity column to link the mapping file with event logs, yielding Table 6.5 and 6.6 for pallet and tray respectively. Then, we split the full sequence of events by grouping events per TsuID and HL step. Table 6.7 and 6.8 show the event logs after splitting traces and group events per HL step. More information can be found in [3].



Pallet Event Log with HL Step

TsuID	Activity	Timestamp	HL Step
Pallet1.1	ToLocation1	01:10:00	PALLET_INBOUND
Pallet1.1	ToLocation2	01:10:10	PALLET_INBOUND
Pallet1.1	ToLocation6	01:10:20	PALLET_STORAGE
Pallet1.1	ToLocation7	01:10:35	PALLET_STORAGE
Pallet1.1	ToLocation8	01:10:36	PALLET_STORAGE
⋮	⋮	⋮	⋮

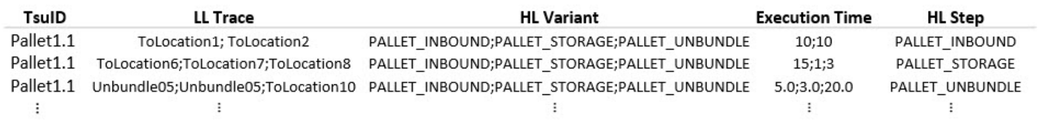
Table 6.5: Example of Pallet Event Log with HL Step



Tray Event Log with HL Step

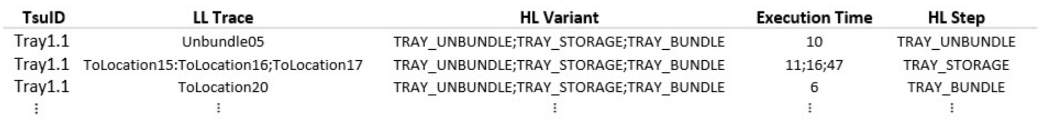
TsuID	Activity	Timestamp	HL Step
Tray1.1	Unbundle05	01:11:00	TRAY_UNBUNDLE
Tray1.1	ToLocation15	01:11:10	TRAY_STORAGE
Tray1.1	ToLocation16	01:11:21	TRAY_STORAGE
Tray1.1	ToLocation17	01:11:37	TRAY_STORAGE
Tray1.1	ToLocation20	01:12:24	TRAY_BUNDLE
⋮	⋮	⋮	⋮

Table 6.6: Example of Tray Event Log with HL Step



TsuID	LL Trace	HL Variant	Execution Time	HL Step
Pallet1.1	ToLocation1; ToLocation2	PALLET_INBOUND;PALLET_STORAGE;PALLET_UNBUNDLE	10;10	PALLET_INBOUND
Pallet1.1	ToLocation6;ToLocation7;ToLocation8	PALLET_INBOUND;PALLET_STORAGE;PALLET_UNBUNDLE	15;1;3	PALLET_STORAGE
Pallet1.1	Unbundle05;Unbundle05;ToLocation10	PALLET_INBOUND;PALLET_STORAGE;PALLET_UNBUNDLE	5.0;3.0;20.0	PALLET_UNBUNDLE
⋮	⋮	⋮	⋮	⋮

Table 6.7: Example of Pallet Event Log with Grouped Traces



TsuID	LL Trace	HL Variant	Execution Time	HL Step
Tray1.1	Unbundle05	TRAY_UNBUNDLE;TRAY_STORAGE;TRAY_BUNDLE	10	TRAY_UNBUNDLE
Tray1.1	ToLocation15:ToLocation16;ToLocation17	TRAY_UNBUNDLE;TRAY_STORAGE;TRAY_BUNDLE	11;16;47	TRAY_STORAGE
Tray1.1	ToLocation20	TRAY_UNBUNDLE;TRAY_STORAGE;TRAY_BUNDLE	6	TRAY_BUNDLE
⋮	⋮	⋮	⋮	⋮

Table 6.8: Example of Tray Event Log with Grouped Traces

Grouping Trace

- **TsuID**

  This represents either pallet and tray ID having lifecycle information.

- **LL Trace**

  LL Trace is the sequence of low-level locations belonging to the same items in particular HL step.

- **HL Variant**

  The list of all HL Steps that items pass through in the whole system.

- **Execution Time**

  This uses the timestamps to calculate time spent, in seconds, from one location to the next location.

- **HL Step**

  This field specifies the HL step that the row belongs to. This is used to determine the rows that fall into the same HL step.

Once we have a table grouping traces by HL step, we then aggregate all items that have identical features of HL step, LL trace and HL variant. Then, we count the number of records with the same features and use it as another feature after aggregation. Also, we recalculate time spent by using the average execution time. Additional features have been augmented:

Aggregating Trace

- **Number of Items**

  This number shows the number of items with the same HL step, LL trace and HL variant, yielded by counting during aggregation step.

- **List of Items**

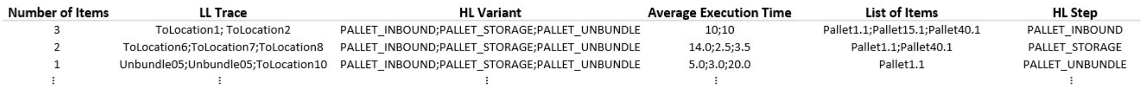
  This list collects all items with the same HL step, LL trace and HL variant during aggregation step.

- **Average Execution Time**

  The execution time is calculated by summing the execution time of corresponding items and then divided by number of aggregated items (arithmetic mean).

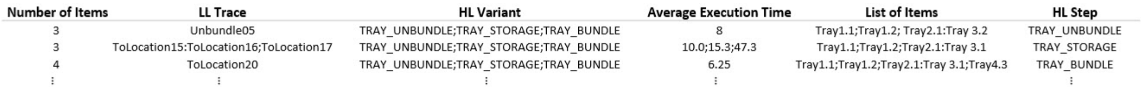
From the exemplified tables (Table 6.7), assume that the trace with Location6, Location7, Location8, belonging to the HL variant "PALLET_INBOUND; PALLET_STORAGE; PALLET_UNBUNDLE" and the HL step "PALLET_STORAGE" (the second row in the table), appears once more in the table with the execution time of 13, 4, and 4 seconds respectively. The TsuIDs of these two items are collected in the "List of Items" feature and the occurrence of two are marked in the "Number of Items" feature. Besides, average time at Location 6 is re-calculated as $\frac{15+13}{2} = 14$ seconds. Similarly, average time at Location 7 and 8 are adjusted to $\frac{1+4}{2} = 2.5$ seconds and $\frac{3+4}{2} = 3.5$ seconds respectively as shown in Table 6.9.

Table 6.9 and 6.10 show the examples on how the actual results from aggregation appear for pallet and tray respectively.



Number of Items	LL Trace	HL Variant	Average Execution Time	List of Items	HL Step
3	ToLocation1; ToLocation2	PALLET_INBOUND;PALLET_STORAGE;PALLET_UNBUNDLE	10;10	Pallet1.1;Pallet15.1;Pallet40.1	PALLET_INBOUND
2	ToLocation6;ToLocation7;ToLocation8	PALLET_INBOUND;PALLET_STORAGE;PALLET_UNBUNDLE	14.0;2.5;3.5	Pallet1.1;Pallet40.1	PALLET_STORAGE
1	Unbundle05;Unbundle05;ToLocation10	PALLET_INBOUND;PALLET_STORAGE;PALLET_UNBUNDLE	5.0;3.0;20.0	Pallet1.1	PALLET_UNBUNDLE
⋮	⋮	⋮	⋮		⋮

Table 6.9: Example of Pallet Event Log with Aggregated Traces



Number of Items	LL Trace	HL Variant	Average Execution Time	List of Items	HL Step
3	Unbundle05	TRAY_UNBUNDLE;TRAY_STORAGE;TRAY_BUNDLE	8	Tray1.1;Tray1.2; Tray2.1:Tray 3.2	TRAY_UNBUNDLE
3	ToLocation15:ToLocation16;ToLocation17	TRAY_UNBUNDLE;TRAY_STORAGE;TRAY_BUNDLE	10.0;15.3;47.3	Tray1.1;Tray1.2;Tray2.1:Tray 3.1	TRAY_STORAGE
4	ToLocation20	TRAY_UNBUNDLE;TRAY_STORAGE;TRAY_BUNDLE	6.25	Tray1.1;Tray1.2;Tray2.1:Tray 3.1;Tray4.3	TRAY_BUNDLE
⋮	⋮	⋮	⋮		⋮

Table 6.10: Example of Tray Event Log with Aggregated Traces

After obtaining the new tables summarizing traces per HL step, trace, and HL variant, we use these tables as interim PCA input.

Following event log grouping and aggregation, we create input matrix for PCA by excluding unvisited locations.

6.1.3 Creating Input Matrix for PCA and Excluding Unvisited Locations

Fundamental concept currently used in Vanderlande to generate input matrix from event log is explained here by using a simplified example in Figure 6.1. This example assumes that the routes shown in the example are all the traces in the system and there are seven locations and two traces occurring; trace 1 visits A,C,D,F and trace 2 visits A,C,E,F while no trace visits location B and G.

Input matrix is created by considering all locations in the system as features. Now, we encode the LL trace as the multiset of locations. In the matrix form, columns represent locations, the rows represent each LL trace,

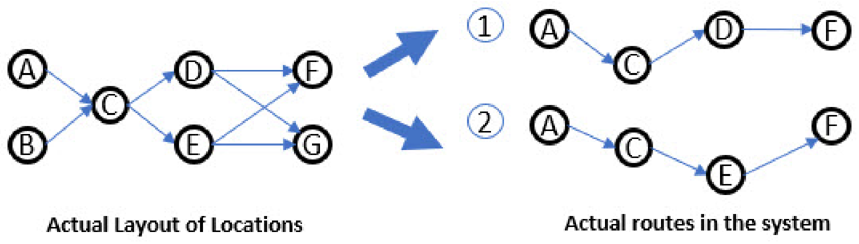


Figure 6.1: Example of Simple Route containing visited and unvisited locations

and the value in matrix is the number of times that the location is passed by the specific trace as shown in Figure 6.2.

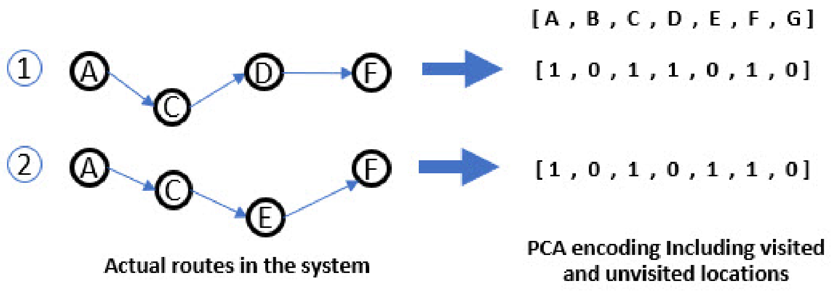

[A , B , C , D , E , F , G]
[1 , 0 , 1 , 1 , 0 , 1 , 0]
[1 , 0 , 1 , 0 , 1 , 1 , 0]
Actual routes in the system
PCA encoding Including visited and unvisited locations
1
2

Figure 6.2: Example of PCA encoding considering visited and unvisited locations

After gaining trace encoding, we multiply the multiset of locations with the number of items (pallets, trays) that take the same LL trace found in the resulting table from aggregation step. Figure 6.3 shows the number of occurrences of each route trace. Trace 1 has three occurrences and trace 2 has two occurrences. As a result, we could gain input matrix considering the occurrences and all locations.

To exclude unvisited locations due to its undersiable impact on the subsequent route clustering, we reduce matrix size by removing the columns (locations) with all zero values. We finally obtain input matrix containing only locations that have been visited at least once as shown in Figure 6.4.

6.1.4 Applying PCA

PCA is performed with the input matrix and the resultant compressed event log is shown in Table 6.5. The current practice on PCA implementation with single-item process makes use of only top three components, allowing visual-

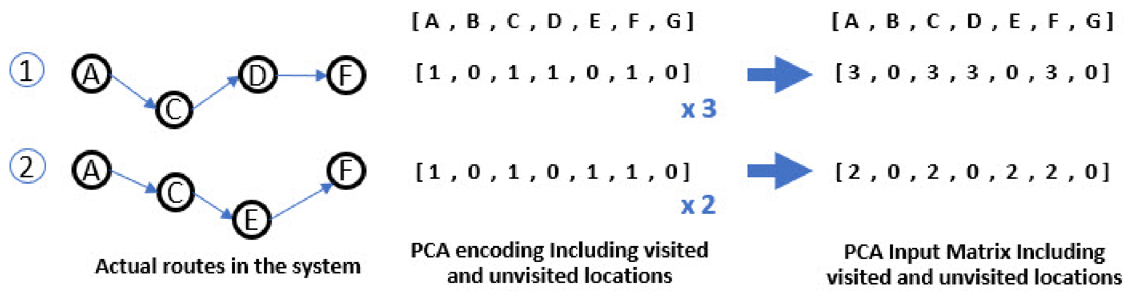

1
2
A
C
D
F
A
C
E
F
Actual routes in the system
[A , B , C , D , E , F , G]
[1 , 0 , 1 , 1 , 0 , 1 , 0]
x 3
[1 , 0 , 1 , 0 , 1 , 1 , 0]
x 2
PCA encoding Including visited and unvisited locations
[A , B , C , D , E , F , G]
[3 , 0 , 3 , 3 , 0 , 3 , 0]
[2 , 0 , 2 , 0 , 2 , 2 , 0]
PCA Input Matrix Including visited and unvisited locations

Figure 6.3: Example of PCA Input Matrix considering visited and unvisited locations

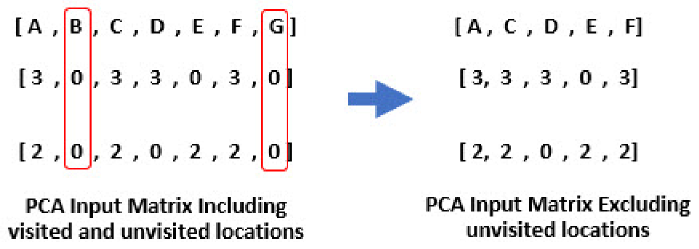
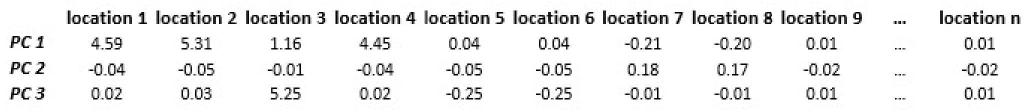

[A , B , C , D , E , F , G]
[3 , 0 , 3 , 3 , 0 , 3 , 0]
[2 , 0 , 2 , 0 , 2 , 2 , 0]
PCA Input Matrix Including visited and unvisited locations
[A, C , D , E , F]
[3, 3 , 3 , 0 , 3]
[2, 2 , 0 , 2 , 2]
PCA Input Matrix Excluding unvisited locations

Figure 6.4: Example of PCA Input Matrix excluding unvisited locations

ization in three dimensions, when analyzing route and performing clustering. Hence, we also use three PCs in our study.



	location 1	location 2	location 3	location 4	location 5	location 6	location 7	location 8	location 9	...	location n
PC 1	4.59	5.31	1.16	4.45	0.04	0.04	-0.21	-0.20	0.01	...	0.01
PC 2	-0.04	-0.05	-0.01	-0.04	-0.05	-0.05	0.18	0.17	-0.02	...	-0.02
PC 3	0.02	0.03	5.25	0.02	-0.25	-0.25	-0.01	-0.01	0.01	...	0.01

Figure 6.5: Example of PCA Output

Afterward, locations are clustered based on similarity in the PCA-transformed visiting occurrences, which will be explained in the next section.

6.2 K-means Clustering

The next step in route summarization is to categorize locations of similarity, as described in Section 2.3.2. K-means clustering technique is here applied on the matrix of PCA-transformed visiting occurrences yielded from the previous step. This allows clustering the locations which are similar according to PCA.

6.2.1 Determining parameter K

Using K-means, it is necessary to pre-define the number of clusters (K) before executing the algorithm. However, the number of locations in the warehouse dramatically varies by HL step, setting global K for all steps would not be suitable necessitating the selection of appropriate K for each HL step, which is done by two complimentary approaches.

First, data are visualized in a space of three dimension, each of which is corresponding to a PC. Data diffusion in this space allows an estimation of appropriate number of clusters by using qualitative assessment. Figure 6.6 illustrates the clusters by number of K from PCA result.

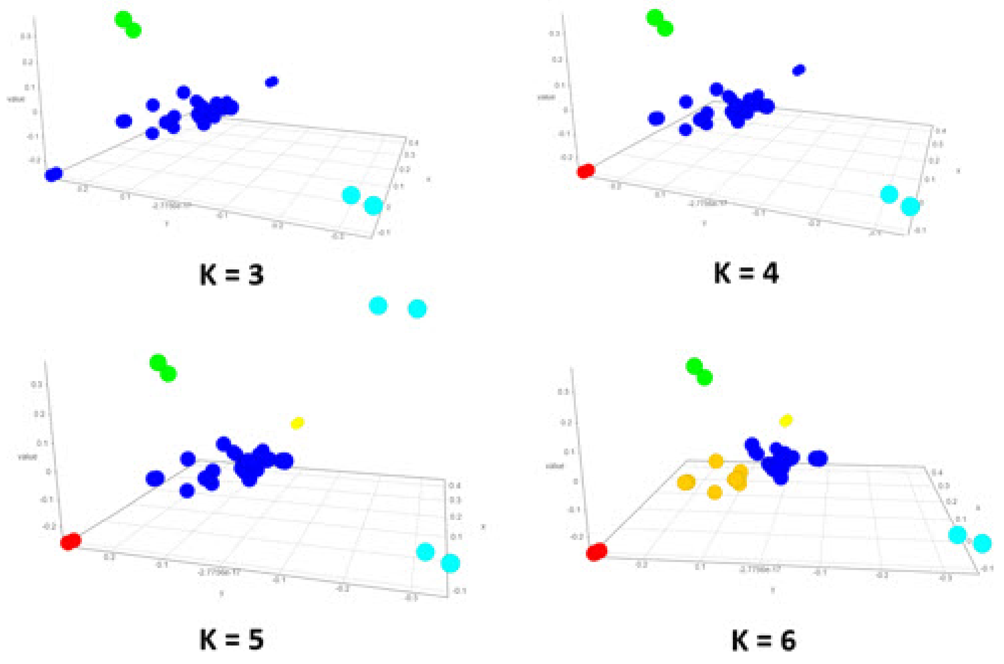


Figure 6.6: Example of K Clusters from PCA 3 dimensions

Second, Euclidean distance of each data point to its centroid is calculated and averaged over all points to be the indicator of how well data are stick into a group (agglomeration analysis). Then, the indicators are plotted against the varying number of K, and the elbow method is used to specify the appropriate K, in a quantitative manner. An example is shown in Figure 6.7, having horizontal axis representing number of K and vertical axis showing the average Euclidean distance.

We use both results from two approaches to determine the suitable number of K for selecting interesting locations. Although the chosen K cannot

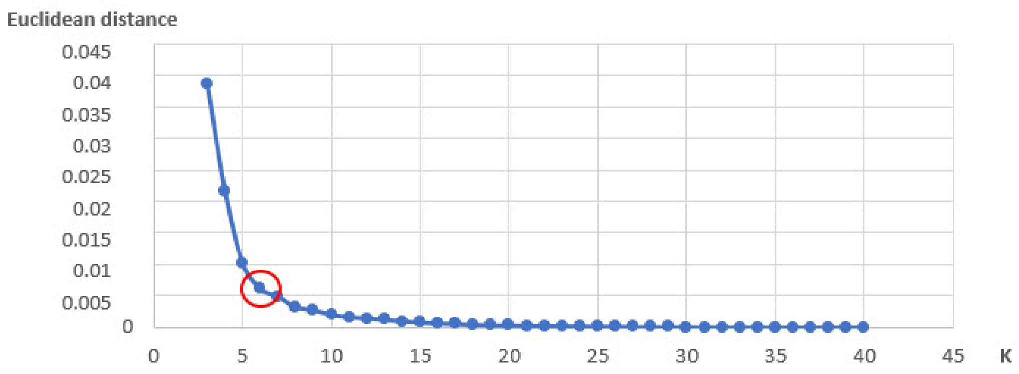


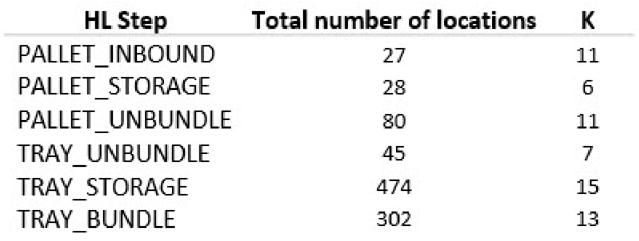
Figure 6.7: Example of the result from Euclidean distance varying by number of K

guarantee the optimal clustering results, it should be a adequately good approximation.

Once we determine appropriate number of K by HL step, we then apply K-means clustering to obtain interesting locations for each of HL steps, these locations are used to create clusters of routes in the system.

6.2.2 Results of K-Means clustering

Table 6.11 shows the selected parameter K for each HL step and the total number of locations. It can be observed that the original numbers of locations vary highly from 27 to 474, suggesting the necessity to select parameter K per HL step. Figure 6.8 shows the average of Euclidean distance between each data point and its centroid, plotted against varying parameter K. The selected K, as highlighted in a red circle, is usually at the elbow of the descending curve. Apparently, K-means clustering can reduce the number of locations to be interested to 6-15 locations.



HL Step	Total number of locations	K
PALLET_INBOUND	27	11
PALLET_STORAGE	28	6
PALLET_UNBUNDLE	80	11
TRAY_UNBUNDLE	45	7
TRAY_STORAGE	474	15
TRAY_BUNDLE	302	13

Table 6.11: The selected parameter K and total number of locations for each HL step

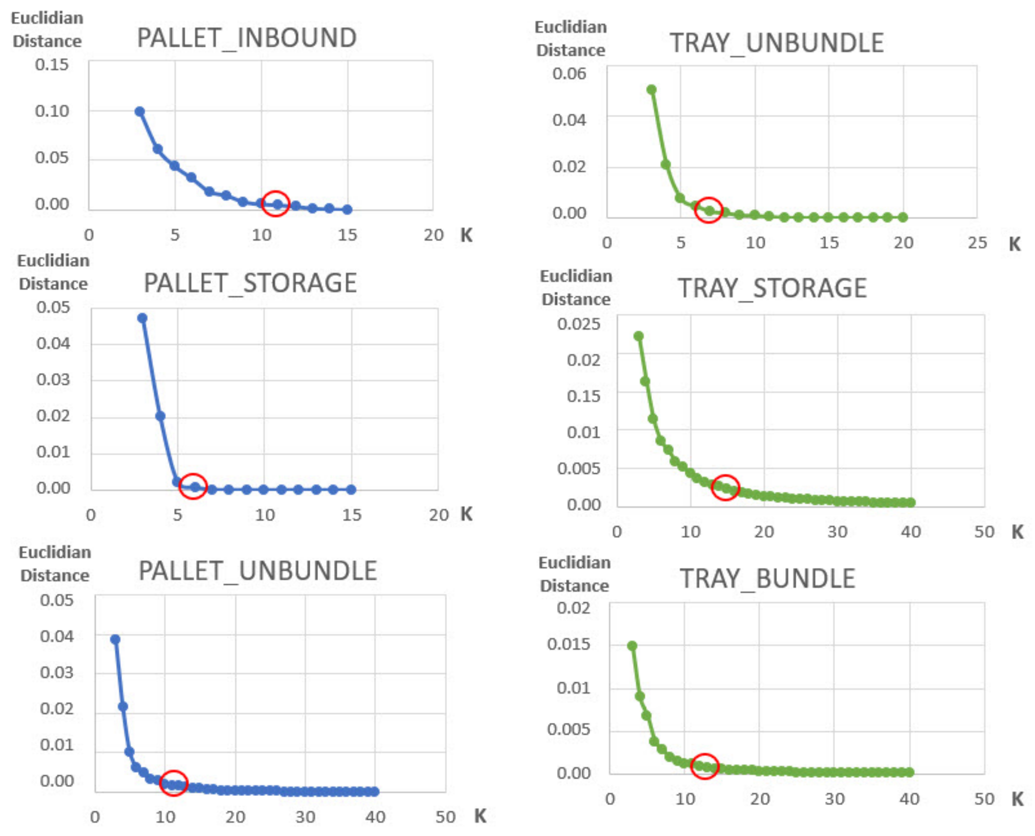


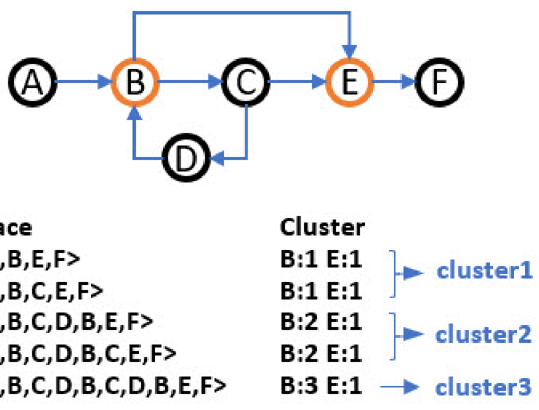
Figure 6.8: The Selected K from Elbow method for each HL step

6.3 Multi-set

The final step of route summarization is to create clusters of routes in WAS that have similar behavior as explained in 2.3.3

After obtaining interesting locations by HL step, we use these locations to create clusters by considering all routes in the event log that transit these locations. Specifically, traces are clusters based on the number of times that the traces pass the locations of interest.

A simplified example is shown in Figure 6.9. Given location B and E as interesting locations, we could see that multi-set abstraction in this example groups Trace1 and 2 into the same cluster because they pass through locations B and E only once for each. Trace3 and 4 are in the same cluster since location B is visited twice while E is visited once. Trace5 is segregated into another cluster because it visits B three times and E only once. If the clustering result is like this, multi-set abstraction of interesting locations is successful considering that it could cluster similar routes and characterize loop and repetitive behavior expected to exist in WAS data.



	Trace	Cluster	
1	<A,B,E,F>	B:1 E:1	cluster1
2	<A,B,C,E,F>	B:1 E:1	
3	<A,B,C,D,B,E,F>	B:2 E:1	cluster2
4	<A,B,C,D,B,C,E,F>	B:2 E:1	
5	<A,B,C,D,B,C,D,B,E,F>	B:3 E:1	cluster3

Figure 6.9: Example of Cluster by applying Multi-set abstraction of interesting locations

The example of final tabular result with cluster information is shown as Table 6.12 and 6.13.

Number of Items	LL Trace	HL Variant	Average Execution Time	List of Items	HL Step	Cluster
3	ToLocation1; ToLocation2	PALLET_INBOUND;PALLET_STORAGE;PALLET_UNBUNDLE	10;10	Pallet1.1;Pallet15.1;Pallet40.1	PALLET_INBOUND	ToLocation1:1
2	ToLocation6;ToLocation7;ToLocation8	PALLET_INBOUND;PALLET_STORAGE;PALLET_UNBUNDLE	14.0;2.5;3.5	Pallet1.1;Pallet40.1	PALLET_STORAGE	ToLocation7:1
1	Unbundle05;Unbundle05;ToLocation10	PALLET_INBOUND;PALLET_STORAGE;PALLET_UNBUNDLE	5.0;3.0;20.0	Pallet1.1	PALLET_UNBUNDLE	Unbundle05:2
⋮	⋮	⋮	⋮		⋮	⋮

Table 6.12: Example of Pallet Event Log with Cluster information

Number of Items	LL Trace	HL Variant	Average Execution Time	List of Items	HL Step	Cluster
3	Unbundle05	TRAY_UNBUNDLE;TRAY_STORAGE;TRAY_BUNDLE	8	Tray1.1;Tray1.2; Tray2.1:Tray 3.2	TRAY_UNBUNDLE	Unbundle05:1
3	ToLocation15:ToLocation16;ToLocation17	TRAY_UNBUNDLE;TRAY_STORAGE;TRAY_BUNDLE	10.0;15.3;47.3	Tray1.1;Tray1.2;Tray2.1:Tray 3.1	TRAY_STORAGE	ToLocation16:1
4	ToLocation20	TRAY_UNBUNDLE;TRAY_STORAGE;TRAY_BUNDLE	6.25	Tray1.1;Tray1.2;Tray2.1:Tray 3.1;Tray4.3	TRAY_BUNDLE	ToLocation20:1
⋮	⋮	⋮	⋮		⋮	⋮

Table 6.13: Example of Tray Event Log with Cluster information

Figure 6.10 shows the number of resultant clusters gained from each HL step. Apparently, clusters of "TRAY_STORAGE" are the most prevalent, yielding 148 clusters in total. However, it seems that clusters of each HL step are not equally interesting, while some may not be worthwhile to further analyze. In particular, clusters of "TRAY_STORAGE" HL step might not provide useful insights for analysis. In addition, a domain expert gave the opinion that knowing the behavior of route in tray storage is relatively less beneficial. This necessitates the assessment of obtained clusters and group insignificant clusters together to suppress their importance and dominance in subsequent analyses.

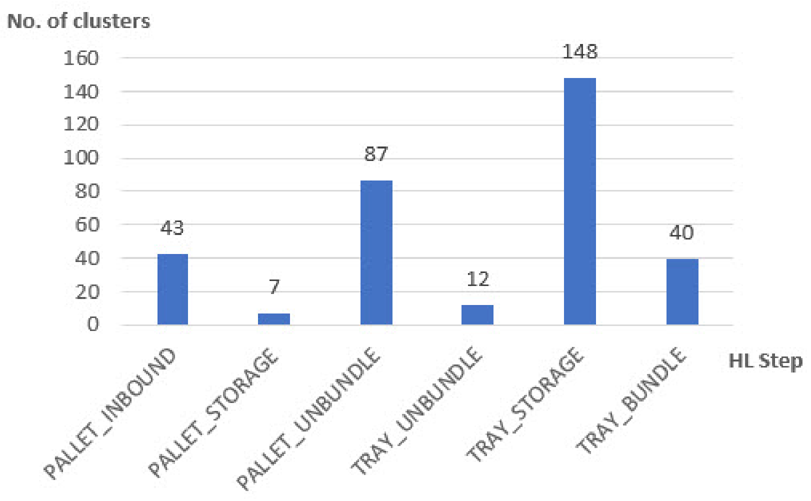


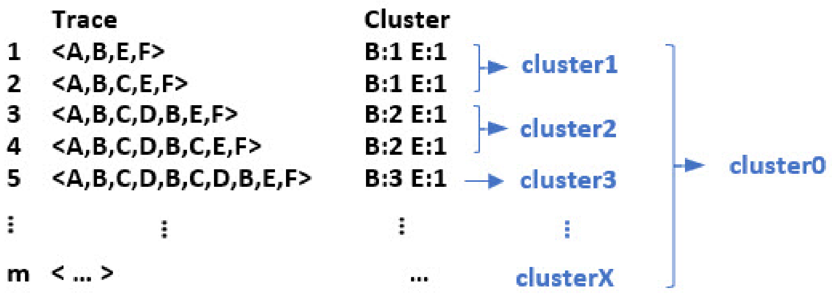
Figure 6.10: The Number of Clusters by HL Step

6.4 Abstracting Clusters

Not all of the clusters are equally interesting. Trivial clusters may hinder the interpretation of system behaviors and should be merged with similarly insignificant clusters to suppress their prevalence in the subsequent visualization. For instance, The trace visiting only locations in "TRAY_STORAGE" should be grouped together, allowing more significant clusters to be more apparent in visualization results, rather than being blurred with the abundant appearance of insignificant routes (which will impede the interpretation). In other words, the results are reorganized based on abstraction.

In this step, we could abstract clusters by relabeling cluster of non-interesting HL step to be a same cluster. Figure 6.11 shows the result from cluster abstraction. Given that the locations in these cluster are not interesting, we relabel all clusters in this example into a single cluster called cluster0.

The result from this procedure is the clusters of routes, worthy for subsequent analysis by visualization that will facilitate route and performance analysis. The procedure is done per each HL step.



	Trace	Cluster
1	<A,B,E,F>	B:1 E:1
2	<A,B,C,E,F>	B:1 E:1
3	<A,B,C,D,B,E,F>	B:2 E:1
4	<A,B,C,D,B,C,E,F>	B:2 E:1
5	<A,B,C,D,B,C,D,B,E,F>	B:3 E:1
⋮	⋮	⋮
m	< … >	…

Figure 6.11: Example of Abstracting Clusters

Chapter 7

Multi-item routes Visualization

By applying the existing single-item technique on multi-dimensional event log, we obtained route summaries for single items for each HL step. We now visualize routes for single item using *single-item visualization*, which is mainly adopted from the existing visualization, and then integrate routes of related items via interrelation table to visualize routes in *multi-item visualization*.

To apply the visualization with warehouse data, it is crucial to define item and lifecycle, pre-process data, and cluster routes as briefly explained in Chapter 4. Afterward, we connect our event log and clustering result with visualization.

Figure 7.1 shows a simplified example of the process with two pallet, five trays, and their relations. It can be seen that Pallet A, Tray 1, 2, 4 and 5 have complete cycles.

7.1 Single-item visualization

As explained in Section 4.3.2, this visualization was originally implemented for a single-item process showing route clustering of each HL variant. We adopt the idea and concept and apply with warehouse data because it is essential to firstly understand each item's behavior. Hence, we aim to find promising characteristics representing the items. In other words, we aim to see notable behavior specifically for pallet or tray in this study.

We first explain the concept for creating single-item visualization. This visualization mainly focuses on an item with a complete lifecycle. When focusing on a single item in Figure 7.1, we focus only on A (complete lifecycle of pallet) and on 1,2,4, and 5 (complete lifecycles of trays) but ignore B (incomplete pallet) and 3 (incomplete tray).

Our event log already contains only pallet and tray with complete life-

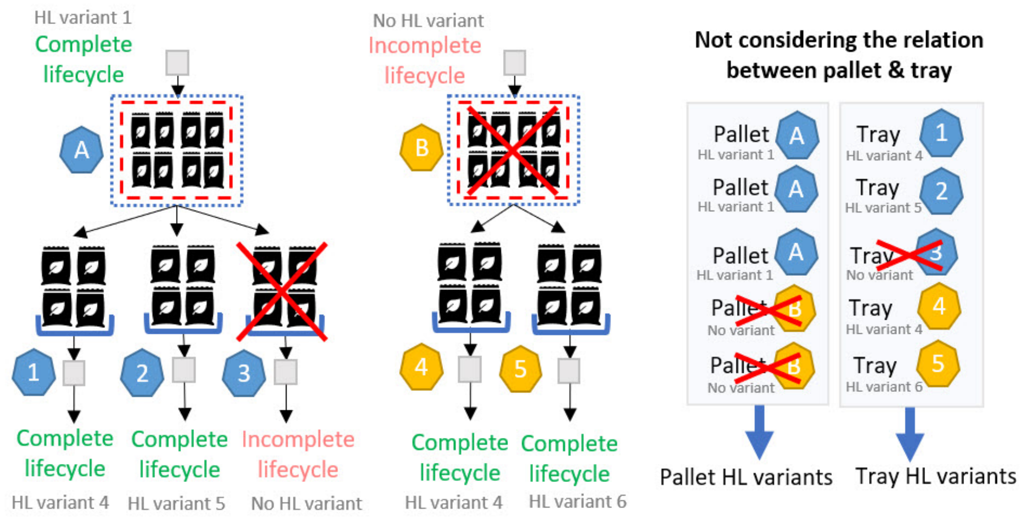


Figure 7.1: Overview of inter-related items to be visualized.

cycle, as defined and explained in Section 5.5, it is hence not necessary to reformat event log prior to creating single-item visualization.

After understanding the concept of the visualization, and how pallet and tray are selected, single-item visualization of a real data can be shown. Figure 7.2 is an example of pallet in single-item visualization having HL variant as PALLET_INBOUND, PALLET_STORAGE, PALLET_UNBUNDLE.

In the visualization, nodes represent locations (start and end locations in each route) and clusters, while edges represent the connections of particular trace to a cluster. Edges are color-coded by the averaged execution time in seconds, where the legend of execution time range is shown in the top of visualization. A HL step consists of start, end locations (darker blue nodes), and clusters (lighter blue nodes); the connection between start node to end node via cluster node represents the route from a starting point to an ending point of the route, and the traverse (in the visualization) via cluster node shows the cluster to which this route belongs. This visualization will be used to analyze unusual behavior of individual items in the following chapter.

7.2 Multi-item visualization

The existing visualization technique, specially developed for single-item processes, cannot demonstrate the interrelation between two or more items. The lack of knowledge on links between item disables the feasibility of an end-to-end analysis to investigate the impact of each item.

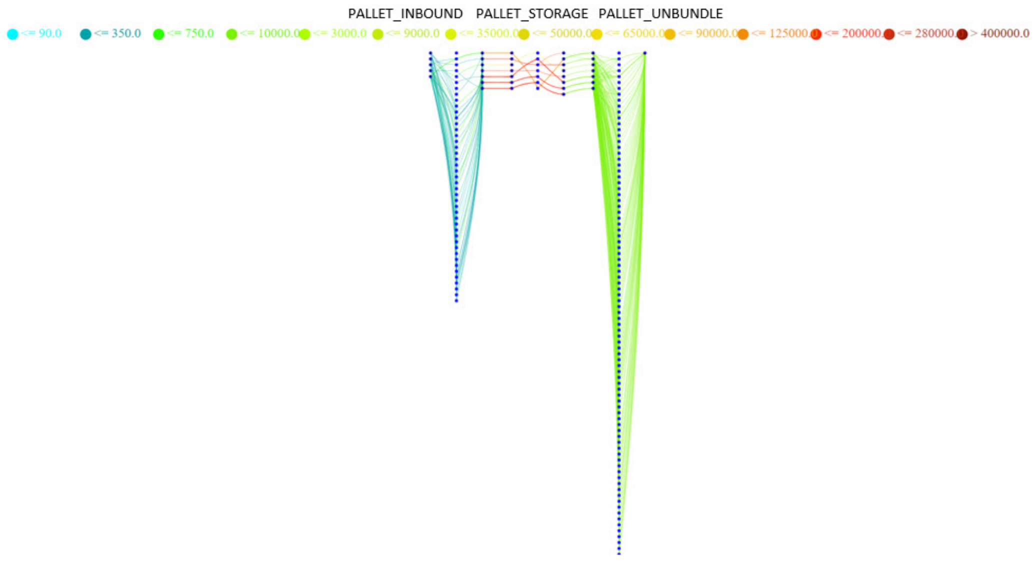


Figure 7.2: Example of Single-item visualization

Therefore, a new visualization technique is developed by extending the capability of a currently available technique.

To deploy the new visualization technique, a pre-requisite for an end-to-end trace is to have complete lifecycles for both types of item (pallet and tray). Partial satisfaction of conditions is not allowed. The requirements for this visualization are:

1. route summaries for items of different type (for pallets and for trays) as obtained in 6.4.

2. interrelation table between items of different type as obtained in 5.4.2.

The following steps are then applied:

1. Filtering pallet and tray from interrelation table to remain only complete pallets and complete trays that have interrelation between each other.

2. Defining appropriate ending points of pallets and starting points of trays.

3. Creating pallet-tray HL variants for each pair of pallet and tray that have a relation appearing in interrelation table.

7.2.1 Filtering

From route summaries, we filter out trays and pallets with incomplete routes (as discussed in 7.1). Then, we further filter to only keep pallets where all related trays have complete routes, and only keep trays where all pallets have complete routes. Finally, we filter to remain only pallets and trays that have interrelation with each other by utilizing the interrelation table, which is capable of representing the distribution of products from a particular pallet to different trays as explained in Section 5.4.

Figure 7.3 illustrates the concept of filtering. In this example, two pallets are considered. Pallet A has complete lifecycle while Pallet B has incomplete lifecycle. Hence, all trays that have interrelation with Pallet B are discarded in this multi-item visualization although both trays have complete lifecycle. Then, we consider Pallet A and its interacting trays. Tray 1 and Tray 2 have complete lifecycle, hence, we include these trays in the multi-item visualization. Meanwhile, Tray 3 has incomplete lifecycle which is filtered out from the visualization. Consequently, we could connect Pallet A with Tray1 and Pallet A with Tray2.

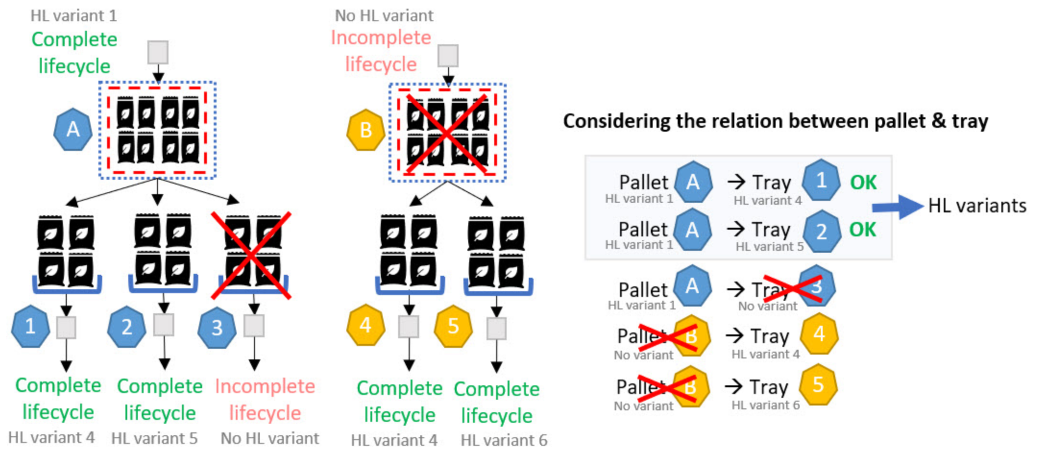

HL variant 1
Complete lifecycle
A
1
2
3
Complete lifecycle
HL variant 4
Complete lifecycle
HL variant 5
Incomplete lifecycle
No HL variant
No HL variant
Incomplete lifecycle
B
4
5
Complete lifecycle
HL variant 4
Complete lifecycle
HL variant 6
Considering the relation between pallet & tray
Pallet A HL variant 1 → Tray 1 HL variant 4 OK
Pallet A HL variant 1 → Tray 2 HL variant 5 OK
HL variants
Pallet A HL variant 1 → Tray 3 No variant
Pallet B No variant → Tray 4 HL variant 4
Pallet B No variant → Tray 5 HL variant 6

Figure 7.3: Concept of Multi-item visualization

7.2.2 Defining Hand-over Point

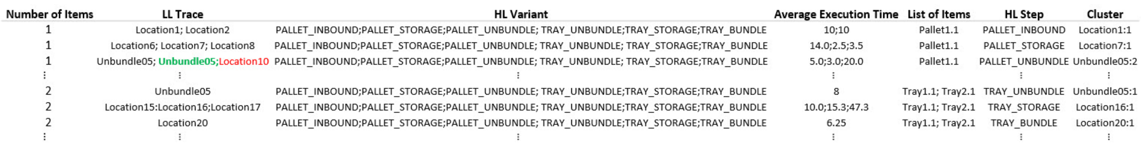
In the new visualization, it is essential that a user defines the appropriate ending point of pallet and the appropriate starting point of tray, apart from selecting only pallets and trays with interrelation. These points are necessary to connect the components to be generated in the next step; one for pallet and the other for tray.

Defining appropriate hand-over points is important for revealing system behavior. Some trivial ending points, such as the state indicating that pool and slave pallets are separated from each other, cannot reveal the dynamics of product distribution from pallet to trays, hindering the interpretation of interrelation. This way, if the trivial points are removed and another meaningful ending point of trays is used, the processes between two variants will be clearer.

7.2.3 Generating integrated visualization

In this step, HL variants of pallets and trays are created simultaneously, under the constraint that pallet and tray have a relation appearing in the interrelation table. Ending point of pallet and starting point of tray are selected from the new pallet-tray HL variant whenever necessary as described in previous step. The resultant event log is used to generate visualization in which pallet and tray are integrated.

To exemplify, Table 7.1 shows event log with new pallet-tray HL variant excluding items without interrelation. In addition, new ending point is defined as an example on PALLET_UNBUNDLE HL step. Instead of having location10 (marked as red in the table) as an ending location shown in the visualization, we select Unbundle05 (marked as green in the table) to be the ending point because this specifies the location and time when products on Pallet1.1 are distributed to multiple trays.



Number of Items	LL Trace	HL Variant	Average Execution Time	List of Items	HL Step	Cluster
1	Location1; Location2	PALLET_INBOUND;PALLET_STORAGE;PALLET_UNBUNDLE; TRAY_UNBUNDLE;TRAY_STORAGE;TRAY_BUNDLE	10;10	Pallet1.1	PALLET_INBOUND	Location1:1
1	Location6; Location7; Location8	PALLET_INBOUND;PALLET_STORAGE;PALLET_UNBUNDLE; TRAY_UNBUNDLE; TRAY_STORAGE;TRAY_BUNDLE	14.0;2.5;3.5	Pallet1.1	PALLET_STORAGE	Location7:1
1	Unbundle05; **Unbundle05;**Location10	PALLET_INBOUND;PALLET_STORAGE;PALLET_UNBUNDLE; TRAY_UNBUNDLE; TRAY_STORAGE;TRAY_BUNDLE	5.0;3.0;20.0	Pallet1.1	PALLET_UNBUNDLE	Unbundle05:2
⋮	⋮	⋮	⋮		⋮	⋮
2	Unbundle05	PALLET_INBOUND;PALLET_STORAGE;PALLET_UNBUNDLE; TRAY_UNBUNDLE;TRAY_STORAGE;TRAY_BUNDLE	8	Tray1.1; Tray2.1	TRAY_UNBUNDLE	Unbundle05:1
2	Location15:Location16;Location17	PALLET_INBOUND;PALLET_STORAGE;PALLET_UNBUNDLE; TRAY_UNBUNDLE;TRAY_STORAGE;TRAY_BUNDLE	10.0;15.3;47.3	Tray1.1; Tray2.1	TRAY_STORAGE	Location16:1
2	Location20	PALLET_INBOUND;PALLET_STORAGE;PALLET_UNBUNDLE; TRAY_UNBUNDLE;TRAY_STORAGE;TRAY_BUNDLE	6.25	Tray1.1; Tray2.1	TRAY_BUNDLE	Location20:1
⋮	⋮	⋮	⋮		⋮	⋮

Table 7.1: Example of Event Log with Pallet-Tray HL Variant for Multi-item visualization; note that Unbundle05 is selected instead of location10.

The visualization is then built by firstly determining the order of item types. In particular, pallet activities occur before tray activities and therefore should be placed on the left side of the visualization. Then, pallet-tray HL variants with the adjustment of ending point of pallet and starting point of tray are used as the input for single-item route visualization (as explained in Section 7.1). In other words, instead of modifying existing program for single-item visualization, we opt to purely adopt the program but with transforming the data into the format that would achieve our design goal to show both pallet and tray with interrelation in the visualization. As pallet part of

HL variant always precedes tray part, pallet visualization thus appears on the left side of the visualization and tray visualization is on the right side. Connections between pallet and tray are generated in the identical way to those between any two adjacent HL steps.

In the visualization, representation of nodes, edges and colors is identical to single-item visualization as explained in Section 7.1. In addition, the connection between ending location of pallet and the starting location of tray is apparent as an edge, which is also color-coded by averaged execution time following the same color ranges as shown in the top legend. Note the 1-1 relation between the ending location of the last pallet HL step and the starting location of the first tray HL step, as the interrelation takes place at the same Unbundle station location.

Figure 7.2 is an example of pallet-tray in multi-item visualization having HL variant as PALLET_INBOUND, PALLET_STORAGE, PALLET_UNBUNDLE, TRAY_UNBUNDLE, TRAY_STORAGE, TRAY_BUNDLE. It is apparent that this HL variant merges both pallets and trays together by their interrelations.

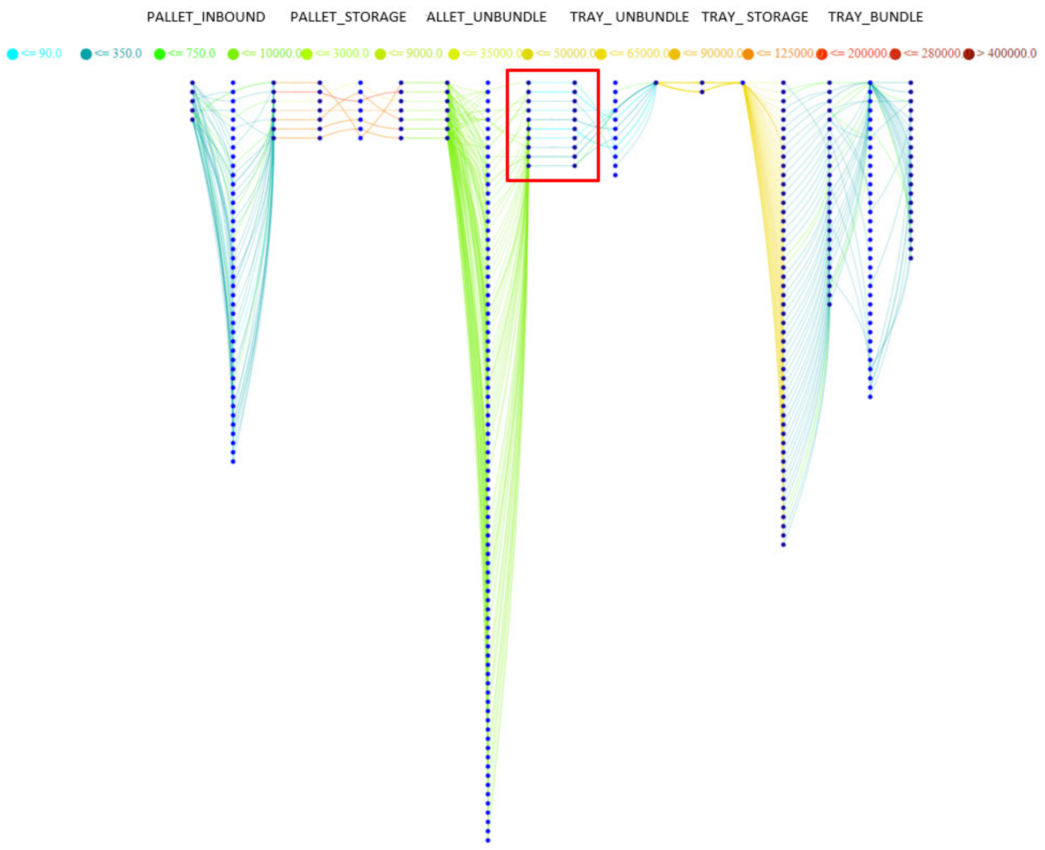


Figure 7.4: Example of Multi-item visualization; note the hand-over points highlighted in red rectangle

Chapter 8

Evaluation

In this chapter, we present the analysis of the technique introduced in Chapter 4. We validate our result by conducting the experiment with warehouse use case, discussing with domain expert, visualizing the result obtained from the proposed techniques. Then, we evaluate and answer our research question and sub-research questions.

We begin the chapter with outlining how to answer research question and sub-research questions by our methods in Section 8.1. Then, we discuss the obtained result and inference from the proposed techniques in Section 8.2 to 8.5.

8.1 Validation of Research Questions

To find the answer to the general research question introduced in Section 1.2 and sub-research questions introduced in Section 3.4, our methods are implemented. We begin with re-introducing our general research question.

Given a dataset of Warehouse Automation System (WAS) and single-item techniques for route and performance analysis, could we adopt and extend the existing techniques to multi-item process to be able to understand the characteristics and performance of inter-related routes?

Then, we evaluate our question by considering sub-research questions and investigate if our proposed solution could answer the questions.

RQ1 *Given a dataset of WAS, how could we understand the behavior of a warehouse and enable the extraction and integration of multiple items and* *their interrelations into a multi-dimensional event log that can represent multi-item dynamics?*

Reading process and system documents and discussion with domain expert, process engineer at Vanderlande could respond to RQ1 by providing more understanding on system behavior and exceptional cases. Beyond system and process comprehension, sub-questions can be responded.

- **RQ1.1** *Given a dataset of WAS, how could we define items, lifecycles and table relations stored in the data?*

- **RQ1.2** *Given a dataset of WAS, how could we obtain route for each item lifecycle including time information?*

- **RQ1.3** *Given a dataset of WAS, how could we obtain interrelation information between items from the source data and interaction information from domain knowledge?*

  RQ1.1, RQ1.2 and RQ1.3 could be answered by applying artifact-centric method and validating the obtained result with a process engineer. With this technique, we could identify items, lifecycle models and relationship among message tables. We use lifecycle model to map start and end activities in the model to the actual event happening in the system. Lastly, the interrelation is defined to connect pallets and trays. Implementation is thoroughly described in Chapter 5.

RQ2 *Given a set of WAS event logs, how could we apply the existing methods such as single-item route analysis, classification techniques and develop a meaningful notion?*

In response to this RQ, we apply previously-proven techniques that are currently used for a single-item process – Airport BHS – with our warehouse multi-dimensional event log as explained in Chapter 6.

RQ3 *Given a set of WAS event logs, how could we explore the behavior of multi-item process flow for a large warehouse and salient flows by textual description or visualization?*

In response to this RQ, exploration is done via the creation of two visualizations as explained in Chapter 7. We adopt the existing visualization used for BHS with WAS and also extend it by creating an end-to-end flows showing the interrelation between pallets and trays.

8.2 Warehouse Multi-dimensional Event Log

8.2.1 Objective

We aim to validate our multi-dimensional event log of warehouse data because this event log is the fundamental input that we use with the adopted techniques for route and performance analysis. We examine if the event log (traces for each item type) is correct with respected to the real system and original data. Hence, domain expert and system documents involve in the assessment of outcome.

8.2.2 Method

As validating each event is infeasible, we therefore focus on the following points.

- Validity of start and end points of item lifecycles: this can be validated by checking redundancy and overlap of start-end pairs of timestamp across lifecycles.
- Uniqueness of identifiers indicating cycles: different lifecycle of item should have unique identifier and this can be confirmed by visual inspection.
- Comprehensiveness of renamed activity: the new name should be from the aspect of executing machine and understandable, where the understandability is assessed by the evaluation from a domain expert.
- Agreement of visualized pattern and actual known behavior: the consistency between visualization of route summarization and system documents is investigated.

In addition, ProM, an extensive framework for process mining [14], is also used to facilitate the validation.

8.2.3 Results

The evaluation suggests that the start and end points of item lifecycles have been verified. No redundant or overlapping start-end pairs of timestamp across lifecycles were found. Pallet and tray event logs are in line with the actual process. Besides, the correctness of unique identifiers indicating cycles has been verified. In addition, the proposed renaming method, which is based on aspect of machine, is qualitatively validated by a domain expert who confirmed the satisfactory comprehensiveness.

Furthermore, the resultant behaviors appearing in the visualized route summaries are in line with system and sub-system design documents, interface requirement specification documents, and the concordance has been confirmed by the domain expert.

8.3 Interesting Locations

8.3.1 Objective

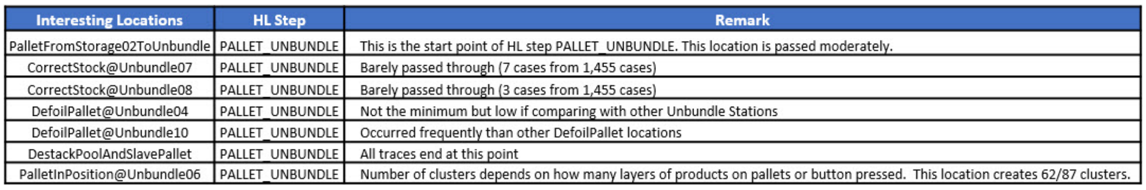
We evaluate the soundness of the result from applying single-item techniques by validating if the obtained interesting locations are sensible with respect to the actual system. In addition, the interpretability of obtained interesting locations is compared with BHS to assess the extent of transferability of the techniques across domains.

8.3.2 Method

We examine the interesting locations of HL step called PALLET_UNBUNDLE. Hereby, we focus at activity and station of the location. Besides, its order in HL step, frequency of occurrence compared with total cases, and the number of clusters that it creates, are taken into account.

8.3.3 Results

Table 8.1 shows samples of interesting locations we used for analyzing the validity. Significant characteristics of each location are included as remarks in the table.



Interesting Locations	HL Step	Remark
PalletFromStorage02ToUnbundle	PALLET_UNBUNDLE	This is the start point of HL step PALLET_UNBUNDLE. This location is passed moderately.
CorrectStock@Unbundle07	PALLET_UNBUNDLE	Barely passed through (7 cases from 1,455 cases)
CorrectStock@Unbundle08	PALLET_UNBUNDLE	Barely passed through (3 cases from 1,455 cases)
DefoilPallet@Unbundle04	PALLET_UNBUNDLE	Not the minimum but low if comparing with other Unbundle Stations
DefoilPallet@Unbundle10	PALLET_UNBUNDLE	Occurred frequently than other DefoilPallet locations
DestackPoolAndSlavePallet	PALLET_UNBUNDLE	All traces end at this point
PalletInPosition@Unbundle06	PALLET_UNBUNDLE	Number of clusters depends on how many layers of products on pallets or button pressed. This location creates 62/87 clusters.

Table 8.1: Interesting Locations of HL step PALLET_UNBUNDLE

- *PalletFromStorage02ToUnbundle*: we found that this location is visited moderately. No significant behavior could be observed from this location.

- *CorrectStock@Unbundle07, CorrectStock@Unbundle08, and DefoilPallet@Unbundle04*: we find that these three locations are passed with small number of occurrences, compared to other locations that execute the same activity but at different stations. At the same time, *DefoilPallet@Unbundle10* is selected because many pallets transit this location more frequently than other stations.

- *DestackPoolAndSlavePallet*: this is the end location of all pallets. Hence, all pallets must transit this location.

- *PalletInPosition@Unbundle06*: this location has a prominent behavior. It creates 62 clusters out of 82 clusters in our result. Then, we further investigate on why this location generates a huge number of clusters, and we consequently found the speciality of this location. We mentioned in Section 3.3.1 regarding the variation of recording method for an event, which we need to keep in mind during performing analysis. This location records activities differently depending on the station connecting to this location. In this case, *PalletInPosition@Unbundle06* is occurred at manual station and it is recorded every button pressing, irrespective of the number of product's layers on pallet. Hence, the number of button pressing is diverse without any fixed number as in automatic station that has maximum 20 layers; this leads to the creation of non-meaningful clusters.

8.3.4 Discussion

According to the result shown in Table 8.1, it can be inferred that while PCA can successfully compress data and preserve important characteristics of process such as frequency of visit, it could also lead to inconclusive results that are difficult to interpret.

In fact, there are considerable differences between warehouse and airport system. A prime example is the number of locations in the system. Airport has much lower number of locations and a set of activities might be mapped toward only one or two locations, such as main and back-up conveyor belt at the same location of the layout. This eases the interpretation of process behavior. In contrast, the same set of activities in warehouse can be mapped toward a wide range of locations in various stations (e.g. there are 11 unbundle stations and 26 bundle stations). This limits the interpretability and is due to the fact that warehouse has many parallel paths that either start or end at the same location. In other words, the captured interesting locations in warehouse system could not directly suggest the distinction of prominent behavior between stations, as they have a same set of activities.

Moreover, the results of finding interesting locations by K-means approach are highly sensitive to the coordinates of location instances in PC space. Only one location closest to the centroid of each cluster is selected with the presumption that this location would well represent the cluster and provides useful insight, which may not always be the case. In contrast, the location that is worth more interest might be located further away from the centroid might not be picked up. This data-driven approach might subsequently yield the route clustering results (by applying multi-set abstraction of interesting locations) that are less useful, especially when interesting locations are not really worth interest.

Although multi-set abstraction of interesting locations in a route is suitable for finding repeated actions or loop, the presence of such behavior might not be prevailing in the context of warehouse or not in the scope of our data. In particular, the repetition of actions might occur with empty tray or filled trays but located in tray storage, which has been discarded in our study as mentioned in Section 3.3.1. Besides, if pallet or tray indeed needs to transit the same location, it often goes to another HL step and comes back later. Hence, repeated activities within HL steps are scarce as shown in the visualization. The limited existence of repeated actions or loop therefore questions the suitability and importance of applying multi-set abstraction in warehouse data.

In conclusion, although PCA, K-means clustering and multi-set techniques work well with airport system due to high interpretability, the cross-domain applicability is doubtful. Especially in our warehouse context, the inference is hard to make due to system layout that has a plenty of parallel paths. In addition, research questions are different across contexts. Nevertheless, the techniques can still provide some general patterns that might be useful for certain analyses.

8.4 Common Route of Each Item

8.4.1 Objective

To evaluate the soundness of HL variants, top three HL variants of pallet, tray, and the merged pallet-tray are investigated, together with the percentage they account for compared to the whole cases. The pattern should be corresponding to the known behavior in the process, therefore a process engineer is involved in the evaluation of the correspondence and providing useful insights.

8.4.2 Method

We select top three HL variants of pallet, tray, and the merged pallet-tray to present current WAS item behavior and their interrelation. The pattern and its occurrence of each HL variant is investigated and discussed with a process engineer to confirm if it is sensible and reflecting system actual behavior.

8.4.3 Results

Pallet HL variants

Figure 8.1 shows the most frequent three pallet HL variants in WAS. The first variant is the simplest pallet characteristic. Pallets enter the system, go to storage and Unbundle Station at the end. In addition, Figure 8.2 illustrates the proportion of HL variants. We could see that first HL variant covers 91.8 percent of all cases while the second and third HL variants combined cover around 5 percent in total.

When we consider the second HL variant, we could observe that pallets enter the system and then go to Clearing Station. After that, they come back to inbound area, go to storage and Unbundle Station similar to the fist HL variant.

Then, we check third HL variant, the route of HL steps are combined the first and second HL variant, connecting by HL variant called PALLET_LIFT.

With this observation, we could infer that a normal pallet tends to have behavior that falls into the first HL variant more than others. Furthermore, the first HL variant seems to be the optimal HL variant of WAS. However, the optimal HL variant cannot automatically imply the optimum of actual physical routes within the HL variant. There are more factors, such as actual point-to-point execution time, number of visited locations and physical distance between locations, needed to be considered when assessing the optimal operation. Our visualization is restricted to HL variant and therefore cannot elaborate the details of LL physical routes. Therefore, the inference from optimal HL variant to optimal LL physical routes cannot be confirmed.

Tray HL variants

Focusing on tray, the top three tray HL variants have interesting behaviors. As shown in Figure 8.3, the three HL variants have highly similar HL steps. The difference is when trays are reused at different bundle stations. Hence, we could see from top three HL variants that trays usually start at unbundle station then directly go to storage and eventually go to bundle station. If trays are not empty, trays are forwarded back to the storage and sent to

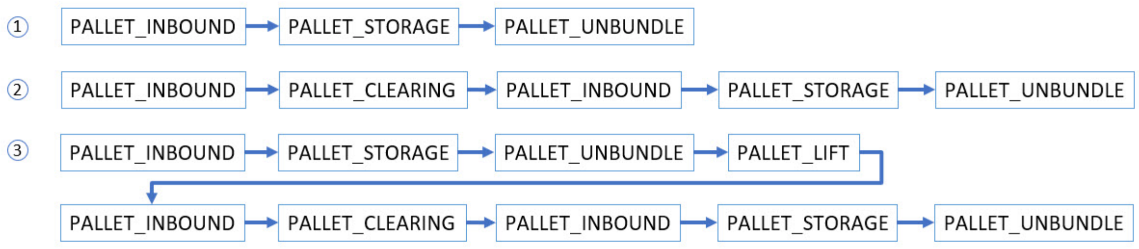


Figure 8.1: Top 3 Pallet HL variants

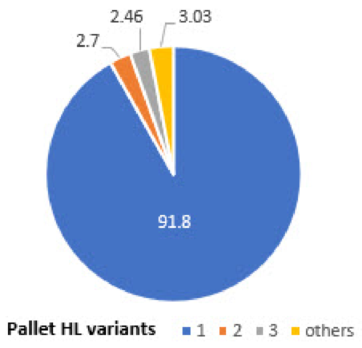


Figure 8.2: Pallet HL variants proportion

bundle station again depending on customers' request. This happens until trays become empty. Trays in the first HL variant are moved to bundle station twice, whereas trays in the second HL variant are used once finishing all the products. The third HL variant visits bundle station for three times until trays become empty.

Then, we analyze the HL variants with their proportion in Figure 8.4, we found that the first and second HL variants have slightly different percentage of the occurrence (around 40 percent). This means that most of trays are used once or twice.

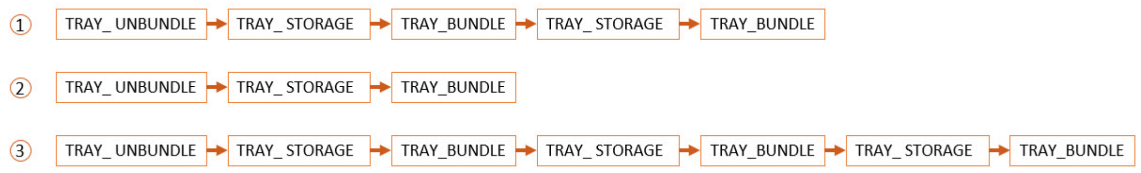


Figure 8.3: Top 3 Tray HL variants

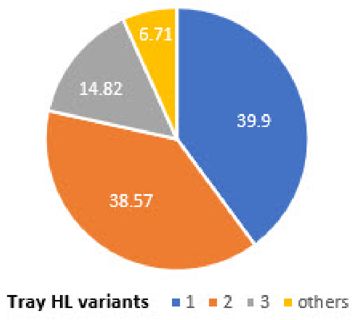


Figure 8.4: Tray HL variants proportion

Pallet-Tray HL variants

Here, we consider end-to-end process. HL variant merges pallet and tray and is called pallet-tray HL variant. Figure 8.5 shows top three of pallet-tray HL variants. We consider pallet-tray HL variants together with the the consideration of pallet and tray HL variants separately as explained in the previous section. It is obvious that the first pallet HL variants (91.8 percent of all pallets) are the first part of all pallet-tray HL variants connecting with top three of tray HL variants.

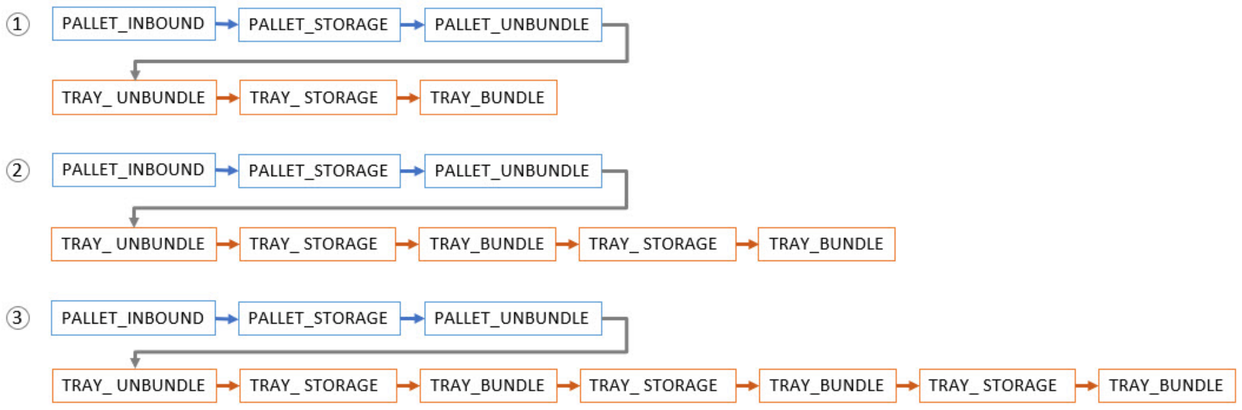


Figure 8.5: Top 3 Pallet-Tray HL variants

8.4.4 Interpretation

Pallet HL variants

We discuss with process engineer about our observation on pallet HL variants. We received the confirmation that the first HL variant is within the

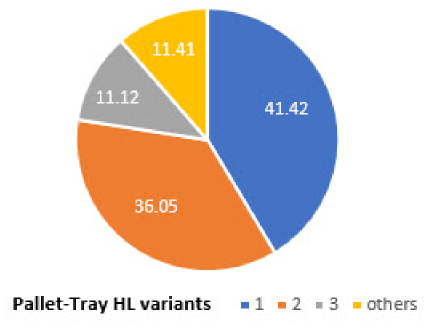


Figure 8.6: Pallet-Tray HL variants proportion

expectation of the process engineer as it takes the shortest HL steps and spends less time compared to other variants. For example, the second and third variants have clearing activity. This activity occurs when pallet has a problem that requires re-measurement, package ordering or other product details which take time to be corrected, and the pallet is then sent back to inbound area.

Tray HL variants

Based on the evidence that most of trays are used for once or twice, we verified result with the process engineer and received the confirmation on repetitive usage. There is no unexpected HL step occurring in our top three HL variants, e.g. TRAY_CLEARING representing incomplete products on tray that requires a staff to manually check case by case. Moreover, the system has been configured with the objective to maximize the use of products in each tray. This is because if trays are not empty, it is necessary to allocate some space in the storage, while the space is very limited in the warehouse. In other words, reducing the number of times requiring the same trays to be present at bundle station can increase the efficiency of warehouse space management.

However, the number of times of using a specific tray is not the only factor to optimize. Size and the number of products should also be considered. Concerning only one aspect in the analysis for the warehouse could not cover all dimensions of system behavior.

Pallet-Tray HL variants

Apparently, the order of tray HL variants in the second part of pallet-tray HL variants are not same as top three of tray HL variants. This is because the percentage of tray HL variant proportion between first and second are close to each other. After we create pallet-tray HL variant, the proportion could be slightly changed due to the additional condition that leads us to filter only trays with complete lifecycle and have interrelation with pallet.

8.5 Interrelation Analysis and Visualization

8.5.1 Objective

After understanding pallet-tray HL variant, next, we consider interrelation between both items. We aim to find an interesting relation that could impact the overall process.

We are specifically interested in the potential delayed operation of tray. As can be seen from Figure 8.3, trays usually start at Unbundle station then directly go to storage and eventually go to bundle station. The arrival of tray at bundle station is purely dependent on the orders of customer, which greatly vary. At this station, a tray delivers products to roll cages (by customer's order) but the time spent in each delivery varies by situations. In normal operation, a tray is effectively used to deliver maximal available products to a roll cage or deliver products to many roll cages in different bundle station without going back to the storage. This usage can minimize the total operation time. On the other hand, a delay can occur when the requested products are out-of-available on the tray, which needs to further request the products from pallets. Delay can also be caused by the placement of tray storage in warehouse; unpopular products might be stored further away from the bundle station, constituting the delay when relocating from storage to bundle station. It can be noticed that the arrival of tray at bundle station can occur for multiple times and the time difference between each arrival can indicate the normal operation and the unusual delayed behaviors. We therefore calculate the time difference between trays coming to bundle station. If tray takes longer time to arrive bundle station than usual, we speculate that the tray encounters a certain problem creating a delay.

The time spent is varying by situations. We regard the following situations as normal time difference:

1. A tray is used for multiple times at the same bundle station without going back to tray storage. In other words, a roll cage in a particular bundle station needs many units of products in the same tray. This means that every time that machine takes product from tray to roll cage, the system records this information. In this situation, the time difference is around 3-10 seconds.

2. A tray is continuously used at multiple bundle stations. This means that many roll cages at different Bundle stations request the same product on the same tray. The time spent is around 25-70 seconds depending on the distance between bundle station.

In the delayed situation, tray might interact with pallet by requesting out-of-available products from pallets as previously mentioned. Therefore, we investigate the interrelation between pallet and tray to examine if the interrelation can provide more insights on the delayed behavior.

After measuring the difference between the arrival of tray at bundle station and knowing whether tray is in normal or delayed situation, we trace back to the pallet that the tray has interaction with by using interrelation table. As a result, we could see the end-to-end flow starting from the time when pallet enters the system to product delivery from tray to roll cage(s).

Hence, we compare two trays that have the same pallet-tray HL variant. One tray has normal time difference (10 seconds), named as normal situation, and the other tray has irregularly long time difference of approximately 30 minutes (2,691 seconds), named as slow situation.

We further investigate the pallets and trays that are associated with a specific roll cage that interacts the trays which are in normal and slow situations. This can be done by creating a new event log as explained next.

8.5.2 Method

The pallet-tray HL variant used for this comparison is the second variant in Figure 8.5. The TRAY_BUNDLE HL step occurs twice in this variant but we analyze only the bundle activity that belongs to our selected tray which is the second TRAY_BUNDLE in this case. Afterward, we compare the flow between these two trays.

Although multi-item visualization allows integrated illustration of the second pallet-tray HL variant with average execution time, this visualization is incapable of showing actual execution time for only particular pallet and trays that are associated with the the roll cage of our interest. It is thus inappropriate in our analysis. We, therefore, propose an alternative visualization, namely *multi-item visualization version 2*, to illustrate actual time spent for specific pallets and trays in separated lines, instead of showing average execution time as in the traditional multi-item visualization. In addition, the programming principle for multi-item visualization does not provide flexibility for modification mainly owing to the technical limitation of the programming function (in file access). The extension of multi-item visualization to achieve our goal is therefore uneasy.

Instead, *multi-item visualization version 2* is rooted from single-item visualization, which allows greater extent of modification. Extended from the previous version that does not consider interrelation between items, we filter only the pallets and trays that are associated with a specific roll cage that interacts the trays which are in normal and slow situations. Then, this filtered event log is used as an input of single-item visualization. As a result, we focus only the first pallet HL variant as shown in Figure 8.1 and the first tray HL variant as mentioned in Figure 8.3 for single-item visualization.

Indeed, we can filter our event log to remain only pallet-tray pairs of interest directly. However, having all pallet-tray pairs for a roll cage could provide us more information for further analysis. This allows a comparison between normal and slow situations and also the pallets and trays within the same roll cages under same HL variant. Consequently, we can see behavioral patterns or overall performance relevant to a particular roll cage in each situation, not only a single pallet-tray pair creating normal or slow situations.

Filtering yields significantly lower number of pallets and trays, providing more space for drawing connections between nodes in visualization. Unlike single-item visualization that merges the number of all pallets or trays with the same start and end points as one single line, our modified visualization shows all lines separately allowing us to specify traces in normal and slow situations.

Moreover, as the single-item visualization shows pallet and tray separately, we need to manually connect both visualizations to illustrate the end-to-end flows and investigate the difference between our normal and slow situations which could be seen in Figure 8.7.

8.5.3 Results

Figure 8.7 shows the resultant visualization. Hereby, nodes represent locations (start and end locations in each route) and clusters. Edges represent the connections among start location, cluster and end location, and color-coded by individual execution time in seconds. Similar to single-item visualization, the connection between start node to end node via cluster node represents the route from a starting point to an ending point of the route in a specific HL step, and the traverse (in visualization) via cluster node shows the cluster to which the route belongs. It should be noted that some clusters are not related to the route that we are considering, yielding no edge connecting the cluster nodes; these clusters are kept for the sake of simplicity but should be ignored.

The yellow-highlighted routes are the routes that have the normal tray and slow tray going to Bundle stations. It can be noticed that there is only one pallet and tray for normal situation, and five pallets and four trays for slow situation under the same pallet and tray HL variant. Note that the numbers of pallet and tray are not required to be equal. One pallet could distribute products to multiple trays and all trays can be bundled to any roll cages under the same or different HL variants. In this example, one missing tray has a different tray HL variant. Hence, we do not show it in this figure as we focus on specific pallet and tray HL variant for single-item visualization.

Next, we conduct our analysis by investigating the significant differences between the situations. We find two prominent differences as highlighted in the red rectangles. Scenario a. encompasses pallet-related route at Pallet Storage. The pallet of normal situation has dark green edges taking around 90-350 seconds while slow situation has orange edges having executing time around 65,000 to 90,000 seconds. The pallet of slow situation stays in Pallet Storage significantly longer than normal situation. Moreover, when investigating other pallets in slow situations, there are also other pallets staying in the pallet storage longer. More routes should be included in a further analysis to confirm the generalizability of this evidence.

Afterward, we consider the second TRAY_BUNDLE in scenario b. and observe that the tray in normal situation goes from Tray Storage to Bundle station at a much faster speed than the tray in slow situation. The tray in normal situation takes around 90-350 seconds while the tray in slow situation spends around 750-10,000 seconds in just relocation. Moreover, we further compare other trays belonging to the same roll cage in the slow situation. We found that only the tray highlighted in yellow takes significantly longer time than others.

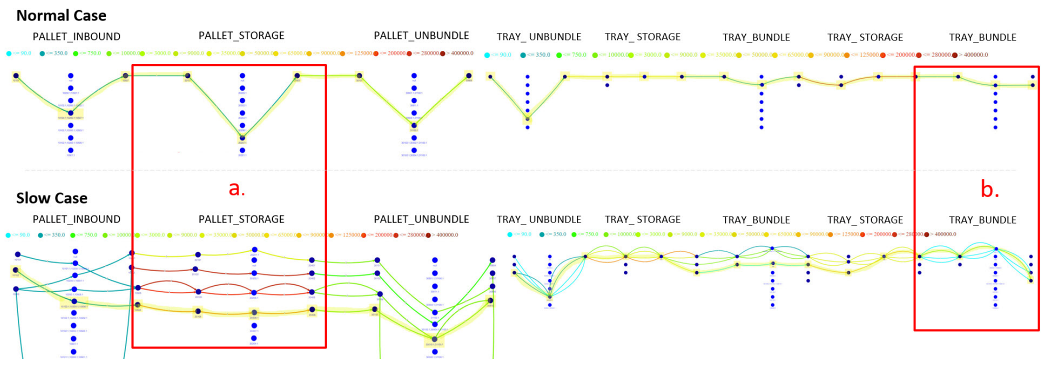


Figure 8.7: Multi-item visualization version 2, comparing normal situation and slow situation.

8.5.4 Discussion

We verify our findings with the process engineer whether these findings create new insights or more information is needed to uncover implicit reasons after applying the proposed techniques. The scenario a. in Figure 8.7, which suggests the pallet in the slow situation takes significantly longer time than normal situation, is informed and could draw the attention of the process engineer evidenced by the returned opinion that the delay might be caused by the activities in Pallet Storage. However, there might be storage management plan at that moment that makes the pallet stays in the storage longer than usual but it may not be the reason that makes tray arrive bundle station late. However, for scenario b. in the same figure, the informed process engineer could not provide strong opinion due to the limited information and the lack of clarity.

Hence, additional information is needed for further investigation. Only time spent in the storage per HL variant may not provide adequate information because we could not explicitly identify or uncover implicit reasons underlying these situations in the investigation on the impact of pallet on tray. Moreover, the visualization obtained from single-item process might not be suitable for interrelation analysis in multi-item process because it could not reveal beneficial information for finding the root cause. Besides, the performance measures that are used to encode color of edges are calculated by average execution time. A modification to allow the visualization to show a separate line for each actual execution time, such as in *multi-item visualization version 2*, is necessary.

To conclude, it can be seen that the existing visualization for single-item process and the extended version are good but not perfect in warehouse context which has interrelation between items. It cannot provide intuitive representation of process and may require some extra work to allow extensive analysis. More importantly, we could not give any solid reason on why particular trays are delayed. We need extra information for interrelation analysis, where investigation only on average time of performance by HL variant is not sufficient. However, both visualizations still provide numerous benefits especially on allowing us to oversee the general system behavior – on which trace pallet and tray flow from start until the end of the process.

Chapter 9

Conclusion

In this chapter, we summarize the outcome of this study. We first conclude the contributions in this project in Section 9.1. Then, we discuss pros, cons and the limitation of our proposed solutions found during conducting the experiment in Section 9.2. Finally, we provide recommendation for WAS in 9.3 and explain the potential future work in Section 9.4.

9.1 Contribution

In this section, we conclude our contributions and elaborate how they could help achieve our research goal. Our contributions include:

- Data Abstraction

  We provide a conceptualization of WAS that has complex un-structured data. We first provide the explanation of warehouse process. Then, we identify key items in the process (pallet and tray), create lifecycle models, and explain the hierarchy of the process both at high and low levels.

- Multi-Dimensional Event Log

  We define the methodology for data extraction. We start from creating unique identifiers and then identify the relation among message tables for connecting data through the whole process, associate lifecycle and the data, and discover the interrelation between items to comprehend work handover between pallet and tray.

- Applicability of Single-item Techniques with Multi-item Process

  We connect multi-dimensional event log with the existing single-item techniques used with BHS in Vanderlande. The techniques used in this study include PCA, K-means clustering and multi-set for route and performance analysis.

- Extended Visualization for Multiple Items

  We adopt the existing visualization from BHS, capable of showing route clustering and performance in different path of the system for one item, and we extend the visualization to be able to show multiple items and their interrelation within one visualization for multi-item process like WAS.

9.2 Discussion

After we analyze the yielded results from our proposed solutions, we could conclude that the artifact-centric approach used to generate multi-dimensional event log works well with multi-item data as in WAS. It facilitates the separation of different items records, creation of end-to-end event log by items, and finding the interrelation between items.

Considering the single-item techniques applied to WAS, these techniques could be used for route and performance analysis. However, it creates difficulties for interpretation in warehouse context. The interesting locations obtained from PCA and K-means clustering are selected in data-driven approach without incorporation with ground-truth knowledge. The speciality of the chosen locations cannot be clearly identified or speculated; it may lie beyond our current knowledge on system behavior, making the results difficult to interpret. The inclusion of these locations when subsequently applying multi-set abstraction considerably affects the consequent clusters.

Next, the number of items to be displayed in the visualization, currently used with BHS, is limited to one. Hence, we could use this visualization to gain more understanding on either pallet or tray behavior but not on the end-to-end process that has both pallet and tray. Consequently, the extension is required to present all items and their interaction in the system. After we extend the visualization, overall behavior of the whole process is apparent. However, it is not suitable for interrelation analysis because the visualization is implemented to understand the routes at high level. Finding the interrelation for some specific pallets or trays is not possible and requires some extra works to be able to perform further analysis such as checking data, filtering event log, calculating time different of tray at bundle station.

9.3 Recommendation

The data recorded in WAS is logged as a communication message between machines. This makes data structure complex with mixed information of many items within each table. With this complexity, it requires inputs from domain expert throughout the whole process of data transformation, investment of time to understand the whole data structure, and performing many manual works to enable finding the relation among tables for a specific system (Dutch supermarket). If studies are conducted on WAS but at a different area from this project, it is necessary re-invest considerable time for data transformation again.

A possible solution is to change the approach that system records data. There are some solutions that could potentially reduce the effort needed for data transformation as follow.

Item Data Separation

The first possible solution is to separate different item records in different table. As the current system mixes pallet and tray records in a transportation message table, if the system could separate this message into two independent tables for pallet and tray, we could reduce the step and time of data pre-processing.

Unique Identifier

One main problem is that items in the system could be reused and the identifiers of these items are unchanged. As a consequence, we could not distinguish the behavior of the items in each cycle. Thus, the system should be able to generate a new identifier when each item enters its new lifecycle.

Non-duplicate Activity Name

There are some activities recorded in WAS having identical name but referring to different actions in the system. With different characteristics, we could not conduct the analysis by using one assumption or single approach while handling data because this could cause a confusion during the analysis. Moreover, statistics (e.g. average, minimum, maximum) derived from this step might be incorrect and lead to misleading results.

A possible solution is to have an identical activity name representing an indistinguishable action. Hence, understanding data could be more straightforward without extensive reliance on domain expert clarifying the activity from different stations, items, and message tables.

9.4 Future Work

In this research, we use various approaches to create multi-dimensional event log and analyze route and performance by applying existing techniques. The results are encouraging future research opportunities that would extend our analyses.

9.4.1 Context Analysis

In this study, we are mainly interested in route and performance analysis by using traces and averaged execution time of different HL step to visualize pallet, tray and their interrelation. We understand the system overview but we could not adequately explain or find underlying reasons for the impact of one item on another item, such as the delay of tray caused by the delay of pallet, handover activities between pallet and tray having an impact on the performance. Thus, relying on two attributes may not be sufficient to determine a root cause, especially for pallet and tray interaction.

Having only a few attributes in the analysis for the warehouse could not cover all dimensions of system behavior. To enable further investigation, it is advised to perform a root-cause analysis by considering another data context. Future research might include additional contextual information such as product description, quantity, package size and dimension. The information might shed a light on the underlying reason why the delay occurred, whether the product creates a different execution time considering its quantity, the number of product layers on pallet or the number of products contained in a tray.

9.4.2 Improving High-Level steps

A certain high-level step can generate overwhelming clusters due to many possibilities of usage. Unlike trivial clusters in Section 6.4, its behavior may still contain meaningful information making it worthy to be analyzed and redefined.

For instance, a tray may be reused for many times as the number of items to be picked up in a single deliver varies by incoming order. The total possibilities of tray usages, calculated from all sub-traces that belong to the same high-level step, create the tendency of gaining excessive clusters that may hinder interpretation.

A possible solution is to perform clustering per each time of tray usage based on the sub-traces yielded from "unrolling" the long trace by the chronological order of tray usage. The diversity of sub-traces is expected to be lower than all original traces, resulting in the lower number of clusters that might be appropriate for finding insights.

9.4.3 Visualization

Current visualization is implemented based on HL variant coming from a collection of chronological HL steps. By showing HL process, the analysis on item interrelations is difficult as the visualization could not show specific pallets or trays of interest. In a future analysis, it would be beneficial to create a new visualization capable of showing LL process. This would enable the capability to specify pallet or tray with significant delayed execution time, thus allowing in-detail analysis to understand the interaction between items and their impacts in WAS.

Bibliography

[1] K. Jensen. Cross industry standard process for data mining (CRISP-DM). `https://commons.wikimedia.org/wiki/File:CRISP-DM_Process_Diagram.png`. Online; accessed 17 July 2020. vii, 4

[2] D. Fahland. *Artifact-Centric Process Mining*, pages 1–10. Springer International Publishing, Cham, 2018. vii, 2, 8, 32

[3] J.M.A. Boender. Context-aware performance deviation analysis for logistics systems. *Master Thesis*, Eindhoven University of Technology, July 2020. 1, 2, 9, 59

[4] Vanderlande Industries. About vanderlande: Company profile. `https://www.vanderlande.com/about-vanderlande/company-profile/`. Online; accessed 20 July 2020. 3

[5] W.M.P. van der Aalst et al. Process mining manifesto. In *Business Process Management Workshops*, pages 169–194, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg. 5

[6] W.M.P. van der Aalst. *Process Mining: Data Science in Action.* Springer, Heidelberg, 2 edition, 2016. 5

[7] W.M.P. van der Aalst. Event log: The input for process mining. `http://www.processmining.org/_media/presentations/event_logs_the_input_for_process_mining.pdf`. Online; accessed 20 July 2020. 5

[8] X. Lu, M. Nagelkerke, D. v. d. Wiel, and D. Fahland. Discovering interacting artifacts from ERP systems. *IEEE Transactions on Services Computing*, 8(6):861–873, 2015. 8

[9] I.T. Jolliffe and J. Cadima. Principal component analysis: a review and recent developments. *Philosophical Transactions of The Royal Society A Mathematical Physical and Engineering Sciences*, 2016. 11

[10] C. Ding and X. He. K-means clustering via principal component analysis. In *Proceedings of the Twenty-First International Conference on Machine Learning*, ICML '04, page 29, New York, NY, USA, 2004. Association for Computing Machinery. 13

[11] W.D. Blizard. Multiset theory. *Notre Dame Journal of Formal Logic*, 30(1):36–66, 12 1988. 13

[12] N. Davies and E. Jokiniemi. *Dictionary of Architecture and Building Construction*, page 410. Elsevier, 2008. 16

[13] K.W. Verhaegh. Process mining for systems with automated batching. *Master Thesis*, Eindhoven University of Technology, August 2018. 27

[14] B.F. van Dongen, A.K.A. de Medeiros, H.M.W. Verbeek, A.J.M.M. Weijters, and W.M.P. van der Aalst. The prom framework: A new era in process mining tool support. In Gianfranco Ciardo and Philippe Darondeau, editors, *Applications and Theory of Petri Nets 2005*, pages 444–454, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg. 79

Appendix A

Acronyms

WAS	Warehouse Automation System
BHS	Baggage Handling System
CRISP-DM	Cross Industry Standard Process for Data Mining
HL	High Level
LL	Low Level
PCA	Principle Component Analysis
RQ	Research Question
LES	Log Extraction Specification
TsuID	Transportation Unit Identifier