Order sequencing and SKU arrangement on a unidirectional picking line.

Jason Matthews

Thesis presented in partial fulfilment of the requirements for the degree of Master of Science (Operations Research) in the Faculty of Science at Stellenbosch University

Supervisor: Prof. SE Visagie

Date: 2011

Declaration

By submitting this thesis electronically, I declare that the entirety of the work contained therein is my own, original work, that I am the sole author thereof (save to the extent explicitly otherwise stated), that reproduction and publication thereof by Stellenbosch University will not infringe any third party rights and that I have not previously in its entirety or in part submitted it for obtaining any qualification.
November 17, 2011

Abstract

An order picking operation in a distribution centre (DC) owned by Pep Stores Ltd, located in Durban, South Africa was considered. The order picking operation utilises picking lines and the concept of wave picking. A picking line is a central area with storage locations for pallet loads of stock keeping units (SKUs) around a conveyor belt. The system shows many similarities to unidirectional carousel systems found in literature, however, the unidirectional carousel system is not common. Sets of SKUs must be assigned to pick waves. The SKUs associated with a single wave are then arranged on a picking line after which pickers move in a clockwise direction around the conveyor belt to pick the orders.

The entire order picking operation was broken into three tiers of decision making and three corresponding subproblems were identified. The first two subproblems were investigated which focused on a single picking line. The first subproblem called the order sequencing problem (OSP) considered the sequencing of orders for pickers and the second called the SKU location problem (SLP) the assignment of SKUs to locations in the picking line for a given wave.

A tight lower bound was established for the OSP using the concept of a maximal cut. This lower bound was transformed into a feasible solution within 1 pick cycle of the lower bound. The solution was also shown to be robust and dynamic for use in practice. Faster solution times, however, were required for use in solution techniques for the SLP. Four variations of a greedy heuristic as well as two metaheuristic methods were therefore developed to solve the problem in shorter times.

An ant colony approach was developed to solve the SLP. Furthermore, four variations of a hierarchical clustering algorithm were developed to cluster SKUs together on a picking line and three metaheuristic methods were developed to sequence these clusters. All the proposed approaches outperformed known methods for assigning locations to SKUs on a carousel.

To test the validity of assumptions and assess the practicality of the proposed solutions an agent based simulation model was built. All proposed solutions were shown to be applicable in practice and the proposed solutions to both subporblems outperformed the current approaches by Pep. Furthermore, it was established that the OSP is a more important problem, in comparison to the SLP, for Pep to solve as limited savings can be achieved when solving the SLP.

Opsomming

'n Stelsel vir die opmaak van bestellings in 'n distribusiesentrum van Pep Stores Bpk. in Durban, Suid-Afrika word beskou. Hierdie stelsel gebruik uitsoeklyne waarop bestellings in golwe opgemaak word. 'n Uitsoeklyn is 'n area met vakkies waarop pallette met voorraadeenhede gestoor kan word. Hierdie vakkies is rondom 'n voerband gerangskik. Die stelsel het ooreenkomste met die eenrigting carrousselstelsels wat in die literatuur voorkom, maar hierdie eenrigtingstelsels is nie algemeen nie. Voorraadeenhede moet aan 'n golf toegewys word wat in 'n uitsoeklyn gerangskik word, waarna werkers dan die bestellings in die betrokke golf opmaak.

Die hele operasie van bestellings opmaak kan opgebreek word in drie vlakke van besluite met gepaardgaande subprobleme. Die eerste twee subprobleme wat 'n enkele uitsoeklyn beskou, word aangespreek. Die eerste subprobleem, naamlik die volgorde-van-bestellings-probleem (VBP) beskou die volgorde waarin bestellings opgemaak word. Die tweede probeem is die voorraadeenheid-aan-vakkie-toewysingsprobleem (VVTP) en beskou die toewysings van voorraadeenhede aan vakkies in 'n uitsoeklyn vir 'n gegewe golf.

'n Sterk ondergrens vir die VBP is bepaal met behulp van die konsep van 'n maksimum snit. Hierdie ondergrens kan gebruik word om 'n toelaatbare oplossing te bepaal wat hoogstens 1 carrousselsiklus meer as die ondergrens het. Hierdie oplossings kan dinamies gebruik word en kan dus net so in die praktyk aangewend word. Vinniger oplossingstegnieke is egter nodig indien die VVTP opgelos moet word. Twee metaheuristiese metodes word dus voorgestel waarmee oplossings vir die VBP vinniger bepaal kan word.

'n Mierkolonie benadering is ontwikkel om die VVTP op te los. Verder is vier variasies van 'n hiërargiese groeperingsalgoritme ontwikkel om voorraadeenhede saam te groepeer op 'n uitsoeklyn. Drie metaheuristieke is aangewend om hierdie groepe in volgorde te rangskik. Al hierdie benaderings vaar beter as bekende metodes om voorraadeenhede op 'n carroussel te rankskik.

Om die geldigheid van die aannames en die praktiese uitvoerbaarheid van die oplossings te toets, is 'n agent gebaseerde simulasie model gebou. Daar is bevind dat al die voorgestelde oplossings prakties implementeerbaar is en dat al die metodes verbeter op die huidige werkswyse in Pep. Verder kon vasgestel word die VBP belangriker as die VVTP vir Pep is omdat veel kleiner potensiele besparings met die VVTP moontlik is as met die VBP.

Acknowledgements

Many people played a significant role in the work leading up to and during the writing of this dissertation. The author hereby wishes to express his deepest gratitude towards:

- Prof SE Visagie for his insight, guidance, enthusiasm and above all his loyal friendship.
- My fellow GoreLab colleges for their friendships, support and technical assistance.
- Dr I Nieuwoudt for her support and friendship.
- My family for their support of over my entire academic career.
- All of the employees at Pep for their friendly and enthusiastic assistance.

The Department of Logistics at Stellenbosch University are hereby thanked for the use of their computing facilities and office space. The financial support of the South African National Research Foundation (NRF), in the form of a Scarce Skills Bursary (with the department of labour) under grant number GUN 20215, of Stellenbosch University in the form of two Merit Bursaries and of Pep in the form of a Bursary towards this research is hereby acknowledged. Any opinions or findings in this dissertation are those of the author and do not necessarily reflect the views of Stellenbosch University or the NRF or Pep.

Table of Contents

List of Figures

List of Tables

List of Algorithms

List of Acronyms

Acronyms	Meaning
ACC	Ant Colony for Clusters
AD	Adjacency Domination variation
AS	Ant System
AS/AR	Automatic Storage and Retrieval Systems
ASA	Ant System Adaptation
CTSP	Clustered Travelling Salesman Problem
DC	Distribution Center
E-GTSP	Equality Generalised Travelling Salesman Problem
FIFO	First In First Out
GP	Greedy heuristic
GPA	Greedy heuristic adapted
GTSP	Generalised Travelling Salesman Problem
HM	Hybrid Method
IP	Integer Programming
LBC	Lower Bound for SLP
LP	Linear Programming
LS	Local Search
MA	Maximum Adjacencies variation
OP	Organ Pipe
OPA	Organ Pipe Adapted
OSP	Order Sequencing Problem
OSPRX	Relaxed Order Sequencing Problem
Pep	Pep stores Ltd
PLAP	Picking Line Allocation Problem
RSC	Random Search of Clusters
SA	SKU Adjacency variation
SAD	SKU Adjacency Domination variation
SKU	Stock Keeping Unit
SLP	SKU Location Problem
SLPCF	SKU Location Problem with Colour feasibility
SLPCFE	SLPCF Exact
SLPCFM	SLPCF Metaheuristic
SSI	Shortest Spanning Interval
TC	Tabu search for Clusters
TSP	Travelling Salesman Problem
VRS	Voice Recognition System

List of Reserved Symbols

Symbol	Meaning	Unit
S_k^i	The span of order k starting at location i	
$\lvert S_k^i \rvert$	The length of span S_k^i	Locations
m	The number of locations in a picking line	
$\lfloor S_k^{\min} \rfloor$	The number of SKUs required by order k	
$\lvert S_k^{\min} \rvert$	The length of a minimum span of order k	Locations
e_k^i	The end location of the span S_k^i	
$\mathcal{N}$	A set of duples (i,k)	
$\hat{\mathcal{D}}$	The set of edges associated with the digraph defined by $\mathcal{N}$	
D	The distance matrix associated with the digraph defined by $\mathcal{N}$	
$\mathcal{F}$	A sub set of $\hat{\mathcal{D}}$	
$\mathcal{C}_k$	A subset of $\mathcal{N}$	
$\mathcal{V}$	A subset of $\mathcal{N}$	
$\hat{\mathcal{D}}(\mathcal{V})$	The set of edges where both vertices are contained in $\mathcal{V}$	
$\delta(\mathcal{V})$	The set of edges between vertices contained in $\mathcal{V}$ and those not contained in $\mathcal{V}$	
$\mu(\mathcal{V})$	$\lvert\{h : \mathcal{C}_h \subseteq \mathcal{V}\}\rvert$	
$\eta(\mathcal{V})$	$\lvert\{h : \mathcal{C}_h \bigcap \mathcal{V} \neq \emptyset\}\rvert$	
x_e	Binary variable for an edge in $\mathcal{N}$	
y_v	Binary variable for a vertex in $\mathcal{N}$	
d_e	The length of edge x_e	
x_{ikl}	Binary variable linking order k starting at location i to order l	
p_k	Position of order k in a sequence	
n	The number of orders in a picking line	
e_{ikj}	Binary parameter for the ending position of order k starting at location i	
x_{ik}	Binary variable assigning order k to start at location i	
C	The maximum of all cuts for the maximal cut formulation	Spans
$\bar{d}_{ikj}$	Binary parameter determining whether an order increases a specific cut	
$\mathcal{S}$	A set of starting positions for all orders	
$\mathcal{E}$	A set of ending positions for all orders	
e	An ending position in $\mathcal{E}$	
s	A starting position in $\mathcal{S}$	
$\mathcal{T}$	A set of subtours	
c_i	The cut associated with location i	

$\bar{C}$	The maximum of all cuts for the revised maximal cut formulation	Spans
$\hat{\mathcal{S}}$	A set of starting positions	
$\hat{\mathcal{E}}$	A set of ending positions	
$\mathcal{A}$	A set of spans	
$\mathcal{B}$	A set of spans	
$\mathcal{C}$	A set of spans	
$\mathcal{D}$	A set of spans	
$\mathcal{G}$	A set of spans	
$\mathcal{A}^*$	A subset of $\mathcal{A}$	
$\mathcal{B}^*$	A subset of $\mathcal{B}$	
$\mathcal{C}^*$	A subset of $\mathcal{C}$	
$\mathcal{D}^*$	A subset of $\mathcal{D}$	
$\mathcal{B}'$	A subset of $\mathcal{B}$	
$\hat{S}^i_k$	The span of order k starting at location i for the relaxed order sequencing problem	
$\hat{e}^i_k$	The end location of the span $\hat{S}^i_k$	
$\hat{C}$	the maximum of all cuts for the OSPRX maximal cut formulation	Spans
$\hat{d}_{ikj}$	binary parameter reflecting whether an order increases a specific cut for the OSPRX	
$\mathcal{M}$	A set of SKUs	
$\mathcal{M}^d$	A set of SKUs which have been duplicated	
ς_t	A SKU	
$\mathcal{P}$	An ordered set of SKUs	
ϱ_i	The ith element of $\mathcal{P}$	
x_{ijkl}	Binary variable linking pick i positioned at location j to pick k positioned at location l	
ς_{tj}	binary variable assigning SKU t to location j	
p^p_i	position of pick i in the sequence	
p^s_j	The SKU in location j	
$\mathcal{O}_i$	The set of picks in order i	
$\mathcal{I}_i$	The set of SKUs corresponding to pick i	
$\bar{d}_{jl}$	The number of locations between locations j and l	
η	The total number of picks for a SLP formulation	
$\mathcal{R}(\varsigma_t)$	The set of orders requiring SKU ς_t	
$\alpha(\varsigma_t, \varsigma_r)$	The number of orders requiring both SKU ς_t and ς_r	
L	An ordered list of SKUs	
ν_{ij}	The visibility between nodes i and j	
$p^k_{ij}(t)$	The probability that ant k places SKU j adjacent to SKU i at iteration t	
α	A parameter controlling the relative importance of the pheromone trail intensity	
β	A parameter controlling the relative importance of the visibility	
$\tau_{ij}(t)$	The pheromone level of edge ij at iteration t	
Q	A parameter of scaling	
ρ	The evaporation factor	
q_i	A cluster of SKUs	

$\oplus$	An operator merging two clusters
m^c	The minimum number of clusters in clustering algorithms
$d(q_i, q_j)$	The distance between two clusters
$\mathcal{R}(q_i)$	The set of orders requiring a SKU in cluster q_i
$\alpha(q_i, q_j)$	The number of orders requiring all SKUs in both clusters q_i and q_j
$\mathbf{z}$	A general solution in a tabu search
$\mathcal{Z}$	The set of possible solutions in a tabu search
$\mathcal{N}(\mathbf{z})$	The neighbourhood of a solution in a tabu search
ψ	A move in a tabu search
$\otimes$	The operator applying a move
$\hat{p}^k_{ij}(t)$	The probability that ant k places cluster q_j adjacent to cluster q_i at iteration t
$\mathcal{U}^k_i$	The set of unassigned clusters
$\mathcal{X}_t$	The set of SKUs similar to SKU ς_t
$\mathcal{C}(q_i, q_j)$	The size of the largest set of similar SKUs present in one of the clusters q_i or q_j
$d^a(q_i, q_j)$	The distance between two clusters in the SLPCF
$d^-(v)$	The in-degree of vertex v
$d^+(v)$	The out degree of vertex v

CHAPTER 1

Introduction

Contents

Logistics involves numerous processes, activities and systems. According to Bowersox *et al.* [5] logistics involves the management of order processing, inventory, transportation, warehousing, materials handling and packing integrated throughout a network of facilities and refers to the responsibility to design and administer systems to control movement of raw materials, work-in-progress and finished products at the lowest cost. It has been formally defined by the Council of Supply Chain Professionals as "the part of Supply Chain Management that plans, implement and controls the efficient, effective forward and reverse flow and storage of goods, services and related information between point of origin and the point of consumption in order to meet customer requirements" [14].

Two possible focal parts of logistics include the effective supply and delivery of raw materials to manufacturing plants or the delivery of partially completed parts from different manufacturing plants for final assembly. In the retail industry, however, logistical functions need to have correct quantities of finished goods at the correct times at retail outlets to maintain customer satisfaction. One of the reasons why logistics exists is to move and position inventory at the right quantity to the right place at the right time in the most efficient and effective way.

1.1 Logistical functions

According to Grant *et al.* [14] there are several functional areas associated with logistics management, namely:

1. order processing,
2. inventory management,
3. transportation,
4. warehousing/distribution, material handling and packaging, and
5. facility network design.

The relationships between these areas may be described using a simple retailer environment as an example. When a customer purchases a product, that purchase is processed at the till and the information passed on to management. The ongoing sales reduces inventory levels and the available inventory must be managed to avoid stock outs. When inventory levels become too low an order must be placed to replenish the inventory. The new inventory would typically come from a warehouse, but may also come directly from suppliers. A warehouse typically consolidates a number of these orders for different retail outlets. The warehouse must put together the orders and distribute them to the retail outlets. This logistical framework revolves around the movement of goods between facilities and therefore the overall efficiency of all of these processes is bounded by the effective placement of all the facilities.

All these functional areas are interdependent, to different degrees, in logistics networks. A brief discussion of these areas as well as their effect on the overall network is given in the following sections.

1.1.1 Order processing

An important step in order processing is to predict or forecast the customer's needs and inventory requirements. Forecasting may take many forms depending on the position of the operation in the network. Marketing departments influence customer demand by means of promotion, pricing and competition. Manufacturing departments, however, forecast production requirements based on sales demand forecasts and current inventory levels.

Forecasting allows for the aggregation of all customer orders for operational and strategic planning, but customer orders must still be handled individually. This requires the processing of order receipts, delivery, invoicing and collection. The order processing functional area interacts directly with the customer and inefficiencies in this area directly effects customer satisfaction.

1.1.2 Inventory management

The inventory levels of a firm depend mainly on the logistical network and the desired level of customer satisfaction. Customer satisfaction may be expressed as the percentage of time a customer is able to purchase goods because the required inventory is available. The greater the required level of satisfaction the higher the level of available inventory must be which results in higher risk due to market fluctuations, stock damage and capital costs. Overall inventory levels may be lowered while still reaching the same levels of customer satisfaction by increasing the efficiency and effectiveness of the logistics network. Two factors might achieve this goal.

1. Decreasing the lead times increases the rate at which stock can be replenished. This reduces the risk of variable demand between deliveries as the time between deliveries are shorter.

2. Increasing consistency of the network reduces the risk of failed information flow and late or incorrect deliveries. Both of which must be compensated for by holding additional safety stock.

1.1.3 Transportation

Transportation refers to the physical movement of inventory between facilities. There are three main factors which determine the performance of the transportation function, namely cost, speed, and consistency.

Cost refers to the explicit cost of moving a certain quantity of product between two facilities. Both the physical cost (such as fuel and labour) of moving inventory between two locations as well as the cost of maintaining the levels of inventory in-transit (*e.g.* insurance) must be taken into account. Speed of transportation is the time required to move inventory between two locations. Typically the faster the transportation the higher the initial costs, however, the cost of in-transit inventory is reduced due to the lower risk. The consistency of transportation refers to the variations in the time required to complete a specific movement of inventory over a number of occurrences and reflects the dependability of the transportation. Consistency is often considered as the most important factor in transportation as inconsistent transportation forces higher levels of safety stock.

When considering a transportation system a satisfactory balance must be found between these three factors of performance. There is a trade off between the explicit costs, speed and consistency, but the implicit cost implications of poor transportation selection may only be realised downstream. For example, retail stores would place a greater value on constancy and speed of transportation as they typically have many deliveries of goods, while a manufacturing plant requiring large amounts of raw materials would prefer a lower cost at reduced speed. All these factors need to be taken into account when deciding on a transportation solution.

1.1.4 Warehousing/distribution, materials handling and packaging

Warehouses or distribution centres (DCs) have many forms and may be used to store, buffer, consolidate, package and ship inventory. Typically a warehouse is used as a central storage facility supplying inventory to a number of smaller facilities. The term distribution centre is used to describe a warehouses which have a stronger focus on the accumulation and consolidation of many products from many suppliers for customers. Both warehouses and DCs differ in terms of functions depending on the industry and position in the logistics network.

An important activity in the warehouse is the handling of inventory which must be received, moved, stored, sorted and assembled or packed for customer orders, be it end users or secondary facilities. One of the risks of handling inventory is that of damage/theft and it is therefore ideal to handle inventory as infrequent as possible.

Another important activity, known as order picking, is the repacking of inventory in order to consolidate different stock items into one package. Here stock items from different suppliers are consolidated for a single customer.

1.1.5 Facility network design

Facility network design is a long term strategic area. It focuses on determining the number and location of all required facilities, (such as manufacturing plants, warehouses and depots) to most effectively service the major customer areas. For example, if the greatest proportion of customers for a firm are found in the metropolitan areas a facility such as a warehouse must be placed in a geographically good position in order to supply inventory to those areas.

1.2 Warehouses and Distribution centres (DCs)

The warehouse may be viewed as that part of a logistics system that stores products (such as raw materials, parts, goods-in-progress, finished goods) at, and between, points of origin and consumption while providing information to management on the status and condition of the products [14]. Figure 1.1 illustrates the position of different warehouses and DCs in a hypothetical logistics network. The most general use for a warehouse is the consolidation and mixing of products from different suppliers and the breaking of bulk orders for customers. Typically there is minimal value added activity in the warehouse although in some cases the assembly of products is performed.

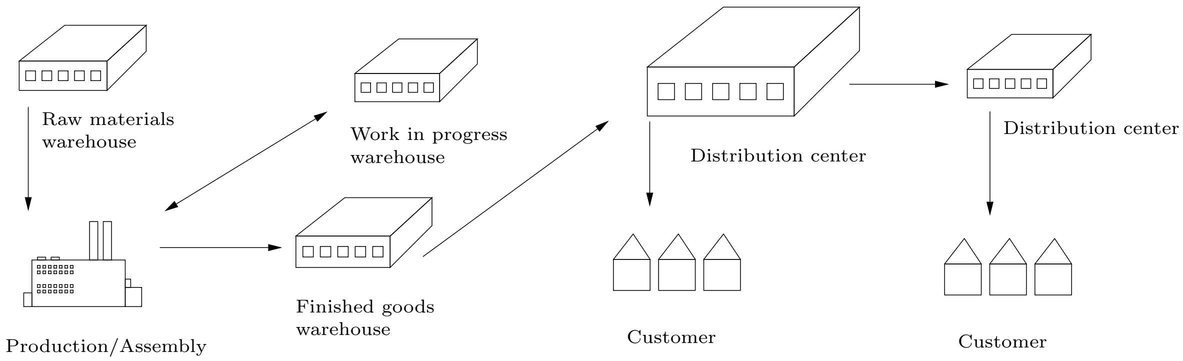


Figure 1.1: *A schematic representation of some possible functional areas in a logistics network where a warehouse or distribution center may be found. The arrows indicate possible stock movement.*

1.2.1 Types of warehouses and DCs

Warehouses may be classified by the function within their logistics network. Frazelle [12] has identified several main distinctions: Raw material warehouses hold raw materials at, or near, the point of use in an assembly of manufacturing processes. For example, construction companies order specific raw materials such as sand or stone from a building materials warehouse. Work-in-process warehouses hold partially completed or assembled products as buffers along an assembly or production line. These warehouses are often found in the motor vehicle industry where different body and interior parts of vehicles are produced by different plants. Finished goods warehouses hold completed goods in order to buffer the effects of variance in demand and production schedules. DCs have a stronger focus on the accumulation and consolidation of many products from many suppliers for customers and are typically found in retail industries. DCs may serve customers directly or serve as an intermediary between suppliers and smaller local DCs or depots.

Further distinctions have been made by Bartholdi & Hackman [2] between different DCs according to the customers which they serve. A retail DC typically supplies products to retail stores such as supermarkets or clothing chain stores. A typical shipment may have hundreds of items and with a large pool of customers and the flow through the DC is high. An example of an organisation using retail DCs on massive scales is the supermarket chain Walmart [34].

A service parts DC typically holds spare parts for expensive equipment, such as motor vehicles, aeroplanes, computer systems, or medical equipment. Due to the types of parts, the demand for an individual part may be hard to predict and if a part is requested it is usually for an emergency. These DCs usually supply spare parts for repairs especially in the motor vehicle industry. This forces large amounts of safety stock to be held on site. A catalogue fulfilment DC usually receives small orders from individuals. Orders are usually 1 to 3 items, but there are high frequencies of such orders that need to be filled and shipped immediately. An example of an organisation making use of large scale catalogue fulfilment DCs is Amazon.com– an organisation specialising in internet sales [1]. A company may outsource all or part of the companies distribution needs resulting in the use of a third party DC. A third party DC can use a single facility to service multiple companies taking advantage of economies of scale and complementary seasonality between two clients.

1.2.2 Warehouse/DC activities

In order to be operational a warehouse requires a number of sequential activities which may be grouped into different functional areas. Frazelle [12] identifies some main functional areas:

- Receiving: All activities involved in the receipt of goods entering the warehouse, providing assurance of the quality and quantity of the goods and dispersing the goods to storage or other functional areas requiring them.

- Prepacking: An optional activity when bulk orders of goods need to be broken down and repackaged into smaller packages.

- Put away: The act of placing goods in storage including product placement, material handling and location verification.

- Storage: The physical storage and record of the position and quantity of goods in the warehouse.

- Order picking: The process of removing individual items from storage to meet a customer order. Activities include the picking of full cartons/cases of goods, individual items or the direct shipping of full pallet loads known as cross docking. This is the main operation around which warehouse designs are based and typically accounts for 55% of total warehouse operating costs [2].

- Packaging and/or pricing: After the order picking operation items may require repricing due to market changes and the items are packaged for easier transportation to the customer.

- Shipping: All activities involved in checking order completeness, consolidating customer orders and loading goods onto trucks or other modes of transportation for delivery.

Figure 1.2 illustrates the relationship between the activities associated with these functions and the flow of goods between them. Material handling is associated with all movements of goods between activities and is central to the operations in a warehouse.

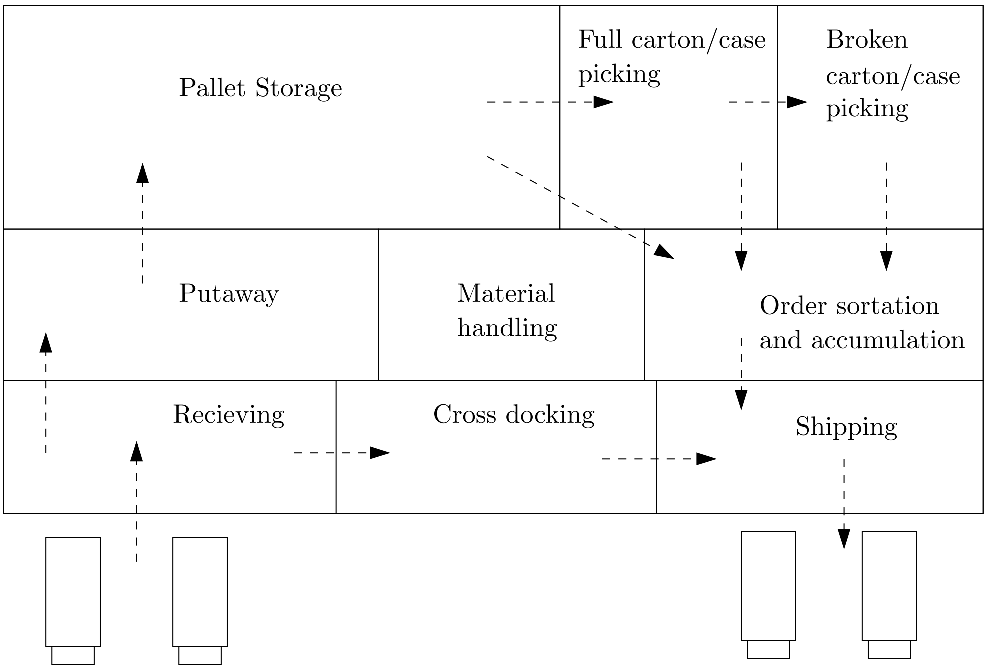


Figure 1.2: *A schematic representation of relationship of different activities in a typical warehouse where arrows indicate the movement of inventory. Material handling is associated with all movements of goods between activities [12].*

1.3 Order picking

Order picking may be described as the process of retrieving products from storage (or buffer areas) in response to a specific customer request [9]. It is the most resource intensive operation usually requiring the majority of the overall workforce and therefore, not surprisingly, accounting for more than 60% of the overall costs in a DC according to Van den Berg & Zijm [33]. Warehouse management systems are therefore usually devoted to the order picking function and many decision support and engineering projects in a DC are associated with this operation.

Due to the differing markets and logistical networks very few DCs run in the same way and use exactly the same order picking systems. Order picking systems may depend on product characteristics (*e.g.* size or fragility) customer order characteristics (*e.g.* frequency and size) and market characteristics (*e.g.* number of customers and customer preferences). It is not surprising then that many order picking systems have been developed and adapted for various needs.

1.3.1 Picking systems

A customer order is the request by a customer for a certain quantity of certain stock keeping units (SKUs) supplied by the DC. The main processes of the order pick include the scheduling of customer orders for processing, assigning specific on hand inventory to the orders, releasing the order to the floor and physically picking the stock from the floor. There are two main types of order picking, full carton/case picking and individual item picking. Full carton/case picking refers to the processing of customer orders where the quantity required of each SKU reflects a number of full cartons/cases as received by suppliers. Therefore the individual SKU can be shipped without leaving the original carton/case. Individual item picking requires the breaking of cartons or the unpacking of cases to consolidate individual stock items. Carton/case picking is much less risky in terms of pick accuracy and theft, because individual items are not handled and quantities are more uniform.

Often many order picking systems may be used in the same DC to manage this process. The two most distinguishable types of systems are automated and manual systems. The majority of DCs run manual systems which make use of human pickers instead of automated machines.

According to De Koster *et al.* [9] the most common manual system is the picker-to-parts system where pickers travel, either by foot or forklift, among the storage aisles in order to pick the required stock. This system may further be distinguished into two types, namely: low-level picking and high-level picking. Low-level picking occurs when all the required stock is within reach from the ground level in the aisle. High-level picking requires a picker to use lifting equipment, such as cranes or forklifts, to lift the picker to the appropriate level in the aisle to pick the stock. The high-level picking is also known as man-aboard picking and Figure 1.3(a) illustrates high level picking where the picker requires equipment in order to reach higher levels of storage.

In a second manual system, known as parts-to-picker system, picking typically uses automated storage and retrieval systems (AS/RS) to fetch stock from storage and bring it to a pick position (or depot) for picking. Once picking is completed the stock is taken back to storage. This system usually utilizes aisle bound cranes or carousel systems.

The third manual system, known as the put or order distribution system, is popular in cases where a large number of customer orders need to be picked in a short time window. This system first retrieves the required stock for all the orders by either making use of the parts-to-picker or picker-to-parts systems. Once the stock has been retrieved it is passed to order pickers who sort the stock into the specific customer orders. A more detailed description of picker-to-parts and parts-to-picker systems will be discussed with examples for the remainder of the subsection.

Picker-to-parts

The picker-to-parts system my be described by the every day task of shopping in a supermarket for a set of items on a list. One simply needs to find all the required items in the supermarket and gather the required quantities of each. Similarly, a picker would receive a list of SKUs with location IDs which need to be picked for an order and the picker would have to find and gather the SKUs in the DC.

The picker-to-parts system may further be described with the distinction between single order/discrete and batch picking. In discrete picking every shopping list, or pick slip, contains only the requirements for a single order. Once that order has been completed the picker receives another pick slip for a single order. If batch picking is used a pick list will comprise of the requirements for a number of orders. The orders are consolidated as a single batch order but must eventually be sorted again into the individual orders. This sorting may be done as the picker picks products by having different bins for each order (sort-while-pick) or the sorting may take place once all the products for the batch have been picked (pick-and-sort). Figure 1.3(b) illustrates a picker in a low level picker-to-parts system with sort-while pick batch picking, where different containers in the trolley contains different customer orders.

Order batching is used extensively in industry and the question which has received much attention is, which orders should be batched together? The order batching problem is a complex problem and many different solution techniques have been applied for different scenarios. Pan & Liu [25] consider the order batching problem in a parallel aisle warehouse. A branch-and-price algorithm was developed in conjunction with a new approximation algorithm for this problem. Hsu *et al.* [17] uses of genetic algorithms to solve the order batching problem by minimising distance and proposes an algorithm which is DC layout independent.

(a) A photograph of a picker using a high level picker-to-parts system. Source: [15].

(b) A photograph of a picker using a low level picker-to-parts system with sort-while-pick batch picking, where each bin represents an order. Source: [23].

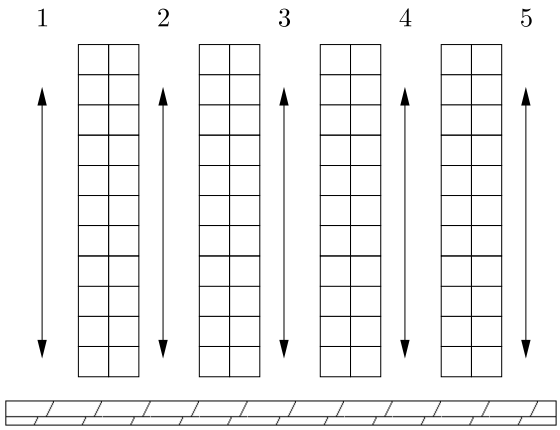
Figure 1.3: *Examples of different picker systems in different industries.*

Another variation of the picker-to-parts system is known as zone picking. Zone picking occurs when pickers are limited to picking only a certain set of SKUs which are geographically close together in the same zone. A picker will only pick the SKUs for a specific order which are present in his zone. Any SKUs outside of a picker's zone must be picked by another picker. A picker therefore only process a part of any specific order. Some benefits of zoning include less travel time, as the operational areas for each picker are reduced, faster pick rates, as pickers become familiar with the products in the zone, congestion minimisation and accountability of picking inaccuracies in each zone [12].

Zone picking can further be split into two categories, progressive and synchronised picking. During progressive zone picking the bins containing the picked SKUs for orders are passed from zone to zone so that the products for each order are consolidated during the picking process. Here pickers pass partially completed orders from one zone to the next for further completion. This creates a situation where downstream zones have to wait for orders from upstream zones to be picked as an order can only be picked at a specific zone once it has been picked in the successive zone. Zone picking can lead to unbalanced work loads and bottlenecks as upstream zones pick faster or slower than downstream zones. A trade off therefore exists between increases in individual pick rates within zones and overall work balance between them. Figure 1.4(a) illustrates a progressive system where each picker is assigned a zone of products and once a picker completes the picks in that zone for an order the bin is passed to the next picker along rollers.

Synchronised zone picking occurs where the products from each zone for an order are picked in parallel and consolidated at the end of the picking line as in the sort-and-pick system discussed earlier. Figure 1.5(b) illustrates schematically the possible zoning within a set of aisles where bins do not get passed between zones but are placed on a conveyor belt destined for consolidation. Pickers do not need to wait for orders from upstream zones but may work at own pace. This does not entirely solve the issue of unbalanced work as an order can only be shipped once all the orders have been picked by all the zones. The unbalanced work is only realised at the sorting activity where the number of partially fulfilled orders may build up. Some investigation into single aisle picking line zoning has been done by Jewkes *et al.* [19]. Jewkes *et al.* uses dynamic programming to assign products to storage locations and partition the aisle into zones. A revolutionary strategy developed by Bartholdi & Hackman [2] known as bucket brigade has, however, addressed both possible balancing issues using a self organising system.

(a) A photograph of pickers in a progressive picker-to-parts system. Source: [10].



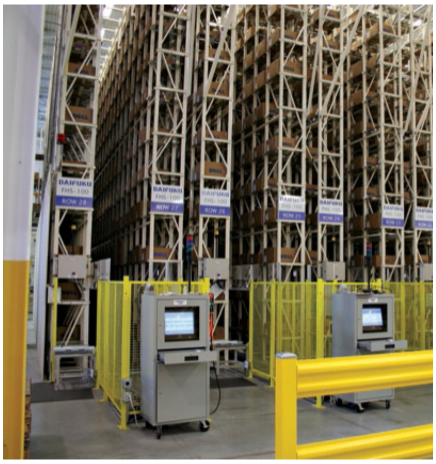
(b) A schematic representation of a possible configuration of 5 synchronised zones within aisles in a DC. A conveyor belt at the bottom of the aisles conveys partially completed orders to the consolidation area. Arrows indicate picker movement.

Figure 1.4: *Examples of different zone picking systems.*

Parts-to-picker

In a parts-to-picker system the picker remains in the same geographical position for the duration of the pick. The physical products are brought to his position and the picker is only responsible for retrieving the correct quantity and not finding the correct SKU. These systems may use different equipment and configurations in order to retrieve the products. Two main systems are automated storage and retrieval systems (AS/RS) and carousels. The parts to picker systems hybridise automation with manual picking. AS/RS systems typically use aisle-bound cranes which may collect one or more pallet or bin loads of product and bring them to the picker or depot. Figure 1.5(a) illustrates a AS/RS system.

Carousels may be seen as a length of shelf fashioned into a closed loop that is rotatable, under computer control, usually in both directions [3]. The carousel presents one or more shelves, each with one or more bins, with products to the picker for picking and is ideal for the storage and retrieval of small parts. Carousels may rotate vertically or horizontally, usually automatically, depending on the DC design and requirements. Hassini [16] gives an overview of carousel systems and discusses different characteristics such as layout, number of parallel stations and rotation direction. Hassini [16] also gives an extensive list of applications of carousels in industry and some solved carousel optimisation problems such as order sequencing and product placement. Figure 1.5(b) illustrates an empty carousel which presents a single shelve with multiple bins to a picker.



(a) A photograph of the control station for a single aisle in a AS/RS. Source: [8].

(b) A picture of an empty horizontal carousel Source: [18].

Figure 1.5: *Examples of different AS/RS configurations and equipment.*

1.3.2 Bucket brigade

One of the complications with zone picking is to balance work between zones and pickers as some pickers pick faster than others and the distribution of products in orders may not be uniform over all zones. A new method of picking introduced by Bartholdi & Hackman [2] known as the bucket brigade remedies this problem. The system may be described as a self organising system and has two main advantages. It requires no planning and it is self adaptive.

The system works roughly as follows. There is one rack which contains all the products (or a sequential system of racks) which needs to be picked and picking for all orders begins at one end (for example, the far left). The fastest picker is given an order and begins the order on the far left of the rack. The next fastest picker then starts an order followed by the third fastest and so on. Once a picker completes an order he places the order on the conveyor (or takes it to the dispatch area) and proceeds to walk back towards the start of the rack (right to left). If the picker comes across another (slower) picker he is required to take the order from the slower picker and continue with that order. The picker which has just handed over his order then walks back until he too finds the previous picker and takes that order until the slowest (left most) picker walks back to start a new order. Figure 1.6 illustrates the steady state of a bucket brigade implementation where the faster pickers naturally pick from larger zones.

The main advantage of this strategy is that the work is evenly distributed relative to the pickers speeds as the faster picker will always have work and gets priority in work allocation. Once the system becomes stable the size of zone served by each picker becomes proportional to the pickers relative picking speed. The system is described as having a pull effect as the fastest picker pulls new orders along to completion. In contrast the progressive zoning approach may be seen as a push system as pickers push work onto the next zone which may overwhelm the next zone or leave the next zone without work.

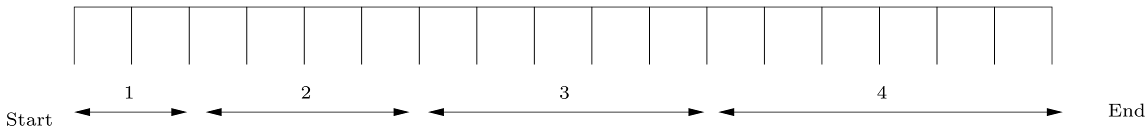


Figure 1.6: *A schematic representation of the stable zoning when bucket brigade is used. The arrows indicate picker movement and the length of the line segments represents the expected zone lengths for each picker.*

1.4 Thesis scope and objectives

A real life order picking system was identified in a DC owned by Pep Stores Ltd (Pep) and forms the focus of this thesis. The scope of the thesis is divided into two parts: Firstly the development of decision support tools for the managers at Pep in order to reduce the time and cost associated with order picking. Secondly the development of tools to be used in future research of the order picking system. This is achieved by perusing the following five objectives.

Objective I

a To describe the layout and operations of the DC so that the problem may be viewed in the broader DC context;

b To describe the order picking system in detail so that the characteristics of the problem may be understood;

Objective II

a Identify long term and short term problem constraints and make suitable assumptions so that a detailed problem may be identified and modelled;

b Identify all levels of decision making in the order sequencing operation;

Objective III

a Make suitable assumptions to model and solve the order sequencing subproblem;

b Make suitable assumptions to model and solve the SKU location subproblem;

Objective IV

a Develop a simulation model to test solution approaches of both the order sequencing and SKU location subproblems;

b Compare results to actual approaches used by Pep;

Objective V

a Discuss potential directions of future studies;

1.5 Thesis layout and organisation

In Chapter 2 the logistical operations of Pep are discussed. A brief description of the broader logistical network is provided as well as a discussion of the DC layout and operations. Furthermore, a detailed description of the order picking operation with focus on the physical process and management systems is given.

The assumptions and problem description is provided in Chapter 3. Both physical and system constraints were identified and suitable assumptions made. The order picking operation is noted to have several hierarchical levels of decision making. Suitable subproblems are defined and the relationships between them are also discussed.

In Chapter 4, 6 and 5 the modelling and solving of the order sequencing and SKU location subproblems is discussed. The simulation model used to evaluate solutions is discussed in Chapter 7. Some results are presented and the solution approaches developed compared to the approaches used by Pep.

Finally, Chapter 8 contains the thesis conclusion including a discussion of possible future work and the use of the presented solution approaches in other studies.

In the following chapter a background discussion of Pep will be given. A brief overview of Pep's logistical network, DC operations as well as order picking operations will be given.

CHAPTER 2

Pep stores background

Contents

Pep is the largest single brand retailer in South Africa with over 1500 stores, in 10 Southern African countries, and has over 15000 employees [26]. Pep is often a lifeline in rural and remote areas where it sells essential items – there is a Pep in almost every town and village in South Africa. It predominantly sells apparel but also sells other products ranging from cell phones to home decór. Pep also owns and runs the largest clothing factory in Southern Africa, which manufactures many of the clothing items sold in Pep. Pep buys merchandise from local as well as international suppliers and is known for keeping profit margins low, due to the low income segment of the market which Pep serves. In order to keep margins low the logistics and distribution systems in Pep are required to be efficient and as a result Pep has won numerous awards in this regard [26].

2.1 Pep's logistics network

Excluding the clothing factories owned by Pep there are three nodes to the logistics network at Pep, namely suppliers, distribution centres and retail outlets. Suppliers consists of both local (in South Africa) and international firms with a distribution operation consisting of 3 main DCs. The largest of these DCs, processing approximately 85% of Peps total stock, is situated at Durban a major port city on the East coast of South Africa with the two smaller DCs situated in Kuilsriver (near Cape Town) and Johannesburg respectively. Figure 2.1 illustrates the structure of the network. The main focus in this thesis will be on the Durban DC as it processes the largest quantity of goods and is the most flexible in terms of operations.

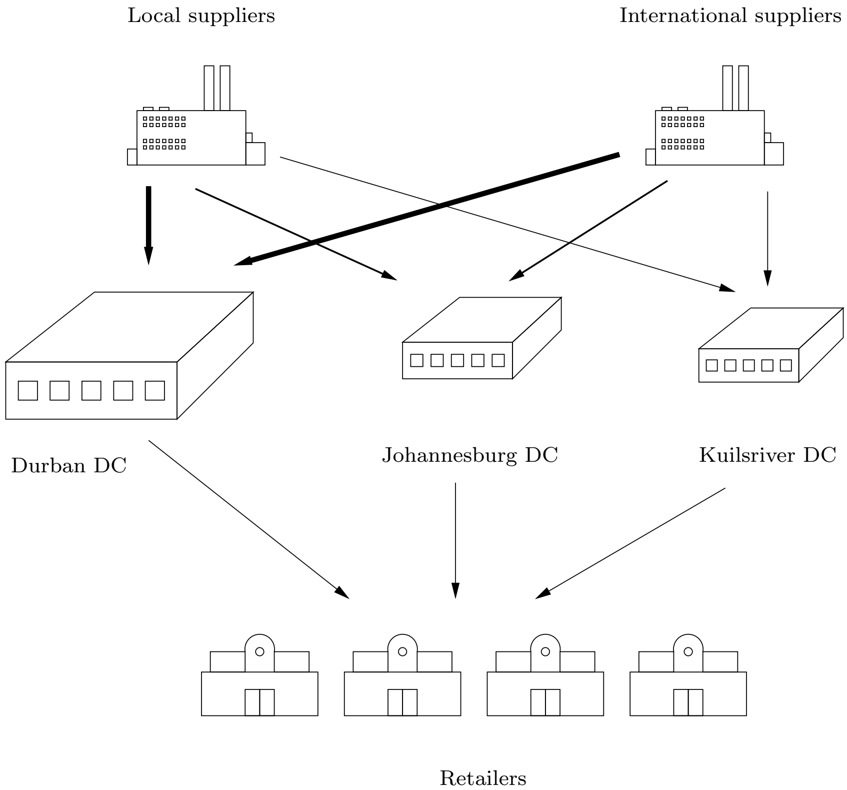


Figure 2.1: *A schematic representation of the fundamental nodes in the logistics network of Pep.*

2.2 Durban DC

The Durban DC, on the East coast of South Africa, receives large quantities of goods from the far East (India and China). The DC is the most crucial part of the entire enterprise as it handles 85% of the stock and forms the most valuable link in the logistics network of Pep.

The storage and order picking operations cover an area of approximately 62200 squared meters and is adjacent to the shipping area of approximately 42776 squared meters. The DC employs roughly 320 employees and processes around 540 containers of stock monthly with operations running on a 24 hour basis.

Figure 2.2 illustrates the physical layout of the DC with areas for the receiving of goods, rack and floor storage and order picking. Specialised forklifts are required to move pallets in the storage racks due to the height of the racks. Pallets are further moved along the floor via standard forklifts or manual pump trolleys. The floor storage is predominantly used for the storage of cartons which can be directly shipped without picking. Figures 2.3 and 2.4 further illustrate the layout and size of the DC storage areas.

2.3 Picking system

Pep has many stores in rural communities and serves a large percentage of the low income population in South Africa. This low income market segment forces Pep to keep profit margins low. Pep, therefore, requires a high turnover of goods in order to remain profitable. Four main

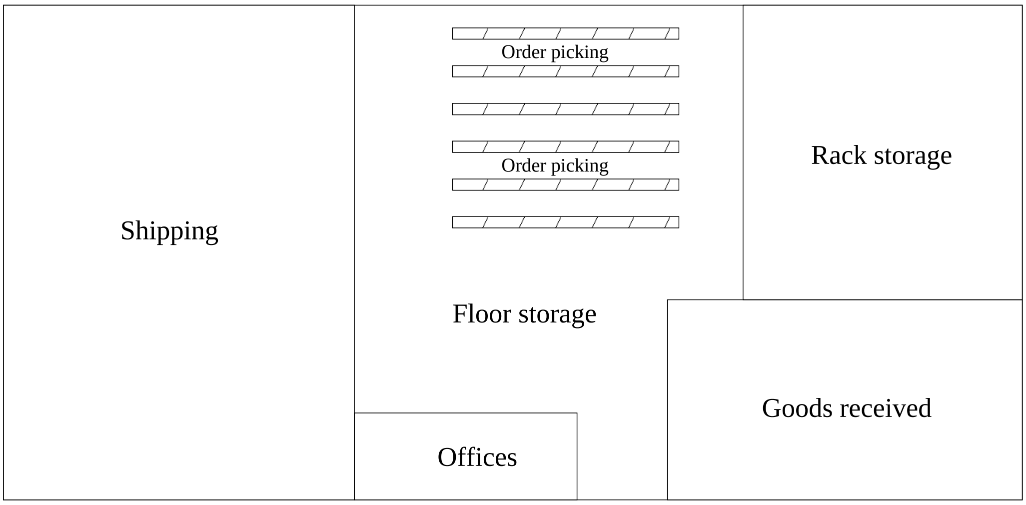


Figure 2.2: *A schematic representation of the layout of the Durban DC.*

Figure 2.3: *A photograph of the floor storage area in the Durban DC.*

factors have influenced the current distribution system in Pep, namely:

- The nature of the products sold by Pep. Clothing items are in most cases bulky items and therefore require large storage areas in the DC.
- The nature of the branches. The DC serves all of the Pep branches each with different product profiles depending on branch location and market segment. However, all branches share a large portion of the base range of seasonal and non-seasonal products.
- The DC has a continuously changing product mix due to the Seasonal and fashion orientated nature of the clothing industry.
- Pep has a philosophy of central planning. Pep limits the number of decisions made by local management and has completely removed any control of stock order from local stores. A central planning department will determine the required stock for each store.

Figure 2.4: *A photograph of the storage racks in the Durban DC.*

These factors have further influenced the overall process of the order picking system, the layout and configuration of order picking areas and the physical picking process. Due to the set list of customers (branches) and the continuous supply of stock to each branch the DC focuses on processing all the branch requirements for a single SKU rather than all the SKUs required for a single branch. The DC therefore processes individual SKUs rather than individual orders. The processing of a SKU requires that all the branch requirements for that SKU are picked and shipped in one operation.

2.3.1 Picking line layout and operation

To allow for easier parallel processing of multiple SKUs the DC uses picking lines which allows for multiple SKUs to be processed locally at the same time. The picking line is an area where individual stock items are placed for the branch requests to be fulfilled. Pallets of goods are placed around a conveyor and pickers move around the conveyor while picking. Figure 2.5 illustrates a typical picking line in the DC.

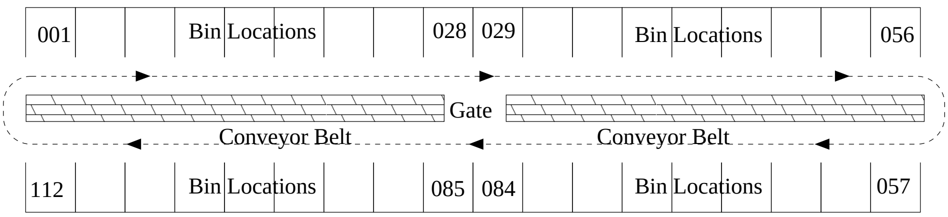


Figure 2.5: *A schematic representation of the layout of a typical picking line in the Durban DC.*

Each location (112 in total) is assigned a single SKU. Due to the size and quantity of the products, usually textiles, a large storage space is required. Multiple pallet loads can be stored at each location as shown in Figure 2.6. Pickers move in a clockwise direction around the conveyor belt doing the order picking as shown in Figure 2.7. Completed cartons are placed onto the conveyor belt which transports the cartons to the shipping area as shown in Figure 2.7. The gate in the middle of the conveyor belt allows for pickers to move from one side of the picking line to the other. This gate may also be used to split the physical picking line into 2 smaller lines of 56 locations each.

Figure 2.6: *A photograph of the rollers which allow for additional safety stock in a picking line at the Durban DC.*

Figure 2.7: *A photograph of a functioning picking line in the Durban DC.*

A picking line operates with the principle that all branch requirements for all SKUs on the picking line must be picked before another set of SKUs may be placed on that line. This style of forcing all the branch requirements to be picked before assigning a new set of SKUs to the picking line is known as wave picking. A wave may be seen as a set of SKUs with all their branch requirements. It may be viewed as the batching of SKUs and not orders. Due to this concept of wave picking an order will refer to the SKU requirements for the SKUs in a wave on a single picking line for a specific branch.

2.3.2 The picking process

The picking process is started by the planning department at central office which will plan branch requirements for each SKU ordered from suppliers. Once the DC has received both the physical SKU as well as the branch requirements for that SKU it may be processed. The DC uses a FIFO (first in first out) system implying that the first SKUs to be received with its branch requirements are the first to be processed.

When a SKU is scheduled to be processed it is assigned to a picking line with typically 55 other SKUs. Usually there are multiple picking lines which become available on the same day. If, for example, 2 picking lines of size 56 SKUs become available on the same day the first 112 SKUs in the FIFO system will be allocated to those two picking lines. The work, or number of picks associated with the 112 SKUs, is then distributed evenly over the available picking lines. This is done by arranging SKUs in terms of density, or number of branch requests, and then evenly spreading the SKUs with high densities over the available picking lines.

Once the SKUs have been assigned to a picking line they are then assigned specific locations within that picking line. The current procedure spreads the high frequency SKUs evenly around the picking line in an effort to avoid congestion between pickers. The SKUs are retrieved from storage and placed in the line using forklifts and pump trolleys. This process of transporting SKUs to the picking line is significantly faster than the actual picking in the picking line. Following the construction of the picking line a team of pickers is assigned to pick all the orders associated with the picking line.

2.3.3 Physical picking

Following the construction of a picking line approximately 8 pickers will be assigned to pick the line, depending on number of picks in the picking line and picker availability. The pickers may only move in a clockwise direction due to space constraints, but may pass each other along the line. The pickers use a voice recognition system (VRS) which relays pick information to pickers, records identification numbers of cartons and requests check digit numbers for accountability and accuracy. Figure 2.8 shows a picker with a headset which is used to interact with the VRS.

When a picker starts picking on a line he places a bar-coded sticker on a new carton and reports the number to the VRS. The VRS then assigns a unique order to the picker. The picker picks all the SKUs in the picking line required for that order and places them in a carton (additional cartons may be used).

The VRS will direct the picker, in a clockwise direction, to the required SKU nearest to the picker. When the picker arrives at the location he has to read a check digit number found at the location and if this number is correct the VRS will return the quantity of that particular SKU required. After the picker informs the VRS that the product has been picked the VRS will direct the picker to the next location to be visited. Once all the requirements for that order are completed the VRS will inform the picker and the completed carton will be placed on the conveyor belt. The VRS then requests a new carton to be prepared and following the reading of the new bar-code number the process will start again until all orders are completed.

In the following chapter the picking line problem is discussed and divided into different subproblems. A detailed description of these different subproblems and the relationship between them is also given.

Figure 2.8: *A photograph of a picker wearing the headset required for interaction with the VRS in a functioning picking line in the Durban DC.*

CHAPTER 3

Problem description

Contents

The order picking process is the current bottleneck in the DC and is managed manually. During the order picking process there are many managerial and planning decisions which need to be made, all of which have an effect on the efficiency of the whole process.

A number of sequential planning decisions have been identified which must be made when attempting to pass a SKU through the order picking process.

1. When should a SKU be placed on a picking line? This is the first decision made once a SKU is received. Currently the DC runs on a FIFO system which was decided on by management. The system forces the SKUs which have been received by the DC to be assigned a picking line first, regardless of any other underlying circumstances.

2. To which picking line should a SKU be assigned? Once the time has come for a SKU to be processed it needs to be assigned to a picking line. When new picking lines are planned there are usually more than one available. This allows for some room to plan the prioritised SKUs, according to FIFO, over a number of different picking lines. Currently the SKUs are assigned to picking lines in such a way that the high frequency SKUs are spread evenly over the available picking lines.

3. Where to place the SKUs within a picking line? Once SKUs have been allocated to picking lines, a line manager decides at which location within the picking line the specific SKUs must be placed. There are a few "in house" rules for this decision. One of these is to

spread the high density SKUs evenly over the picking line in an attempt to avoid picker congestion.

4. In what sequence should orders be picked? As pickers complete orders the VRS assigns new orders to the pickers. A managerial decision was made to use a fixed list system for this order allocation. Management creates a fixed list of orders each associated with a branch. The VRS sequentially moves down the list of orders whenever an order is required by a picker in the picking line. This system can only predict which orders will be picked first collectively by the pickers but does not consider the orders processed by individual pickers.

These decisions are made based on experience and intuition with major emphasis placed on the management of these processes, and not on optimisation. Good decision support systems for these decisions is essential for efficient management. The problem considered here may be described as offering decision support to Pep for the above mentioned decisions to increase the rate of SKU processing while complying to major managerial and physical constraints.

3.1 Constraints and assumptions

There are a number of limitations to decision making encountered in the Durban DC with some being more influential than others. These limitations or constraints may be divided into three groups, structural constraints, managerial constraints and those imposed by the VRS.

3.1.1 Structural constraints

The most binding of the layout constraints is the fact that there is a fixed number of picking lines each with a fixed number of bay locations. Each picking line has a gate in the centre of the conveyor belt allowing for two cases, either a single large picking (112 locations) line or two smaller ones (56 locations). Therefore one cannot simply add additional picking lines in order to speed up the overall process.

Another key issue is work balance in the DC. The DC is managed in such a way that for every functioning picking line there is another picking line being built. This firstly balances the work flow for the forklifts which construct the different picking lines as there is a consistent daily level of forklift work within the picking lines. Secondly, there is an added advantage of always having a wave of SKUs available for pickers who have just completed the picking on an old wave on another picking line. The picking lines of length 112 locations have therefore been split into 2 picking lines of 56 location each.

The final constraint is caused by the pickers themselves. Due to the lack of space one cannot simply flood a picking line with pickers to increase the pick rate as congestion may take place when several pickers require the same product at the same time. Pep has decided on 8 pickers per picking line of 56 locations based on trial and error and observation.

3.1.2 Managerial constraints

The main limitations arising as a result of managerial procedures affects the allocation of SKUs to picking lines and the allocation of SKUs to locations. The first limitation is that a SKU may only be assigned to a single picking line and all the branch requirements for that SKU must be satisfied on that picking line. In addition, the DC uses a FIFO system to allocate SKUs to picking lines. In addition one of the KPIs for the DC requires that all SKUs are picked within 5 days of receiving both the SKU and the branch requirements from the central planning office. Once SKUs are allocated to a picking line certain SKUs may not be placed adjacent to each other depending on several characteristics including size and colour. These adjacency constraints are made to improve pick accuracy as it was noted that when the pick face becomes untidy and new cartons need to be opened the SKUs may get mixed up.

3.1.3 VRS constraints

The VRS imposes many constraints on the problem. The VRS has a master list of orders for a particular line and simply selects the next available order form the list in sequence for the next available picker. This process causes pickers to receive random branch orders depending on the speed and number of other pickers in the line. The second constraint is the allocation of multiple bay locations to the same SKU. The VRS requires that every SKU be allocated a single/unique location in a picking line. Currently SKUs are allocated two adjacent locations if additional storage space is required. The VRS, however, will recognise it as one location with stock and another location without stock. When referring to the concept of allocating multiple locations to a SKU it will refer to allocation multiple non-adjacent locations to the same SKU.

3.1.4 Project scope

After consulting with the managerial staff at Pep the following assumptions were made in conjunction with the managerial constraints:

1. It is assumed that a SKU may only be allocated to a single picking line. Allocating SKUs to multiple picking lines requires a revision of the current planning and warehouse management systems which is deemed as a long term decision. Investigation into this possibility therefore falls outside the scope of the thesis.

In order to handle the structural constraints the following assumptions are made:

2. The number of picking lines in the DC and the size of each will remain unchanged and no consideration into a good mix of picking lines will be considered.
3. It is assumed that the sizes of the picking lines are fixed. A picking line with 112 locations may only be split into two smaller picking lines of 56 locations each. Although the cost and time needed to move a gate is relatively insignificant it is still a long term decision to restructure a picking line and therefore changing picking line sizes falls outside the scope of the thesis.
4. The problem of how many pickers to assign to a picking line falls outside the scope of the problem. The number of pickers working in a picking line is therefore limited to a maximum number which is dependant on the number of bay locations in the picking line. The number of pickers is determined by management based on managerial experience and is usually 8 per wave.
5. The time required to construct a picking line is assumed to be significantly less than the shortest possible time to complete a full picking line. It is therefore assumed that the building of a picking line will never hold up the order picking process.

Although the VRS constraints are currently binding, after consulting with the mangers at Pep it was decided that most of these constraints may be ignored as a revision of the VRS may be done if necessary. The following assumptions were made with regards to the VRS:

6. It is assumed that the VRS can select any pending order to be passed to a picker.
7. It is assumed that a SKU may have multiple locations within a picking line. This assumption is made to determine if any advantage exists from duplicating a SKU over multiple locations in the same picking line.

To determine if a cost exists in restricting allowable adjacencies two cases will be considered:

A The case where any two SKUs are allowed to be adjacent on a picking line.

B The case where certain SKUs identified by Pep are not be allowed to be adjacent on a picking line.

3.2 Problem deconstruction

The problem may be divided into three levels of decision making, the allocation of SKUs to picking lines, the positioning of the SKUs within its picking line and the sequencing of the orders. These three decision tiers may be viewed as three separate but interdependant subproblems, namely:

1. **Picking line allocation problem** (PLAP): In this subproblem SKUs are assigned to picking lines. This problem also includes the decision of whether or not to allocate additional bay locations to SKUs within the same picking line.
2. **SKU location problem** (SLP): Here each SKU is assigned to a location within a picking line.
3. **Order sequencing problem** (OSP): In this subproblem the sequence of the orders for respective pickers within a picking line is determined. This problem considers the manner in which the VRS assigns new orders to pickers.

All three subproblems have the same objective of minimising the time required to pick all SKUs. The decisions associated with each subproblem are made sequentially by first assigning SKUs to picking lines, then positioning the SKUs on their respective picking lines and finally scheduling orders for the pickers on the functioning picking line. The subporblems cannot, however, be investigate and/or solved in this order. It must be approached in reverse order. Before the correct positioning of SKUs in a picking line may be investigated a correct order sequencing procedure must be established with which to evaluate the SKU positioning. These subproblems will thus be investigated in reverse order.

3.2.1 OSP

The first subproblem which must be considered is the OSP. The problem may be described as the sequencing of all the orders in a wave, for the pickers, for a given picking line with fixed SKU positions such that the total picking time is minimized.

For each order there is a subset of the SKUs in the picking line which must be picked. Each picker must first complete an entire order before moving on to the next order. Once a picker completes an order the VRS will inform him that a new order must begin and direct the picker to the next required SKU for that order. Pickers are required to move in a clockwise direction around the conveyor belt and may pass each other. This subproblem is further discussed in Chapter 4.

3.2.2 SLP

The next subproblem focuses on assigning the SKUs to locations in the picking line so that the total picking time is minimised. The SLP requires a measure by which SKU configurations may be compared and therefore relies on the solution approaches to the OSP. This subproblem is localised to individual picking lines and is encountered only once SKUs have been allocated to the picking lines.

This problem must take into account minor managerial constraints and must be able to handle the inclusion of duplicated SKUs. Although within the current system a SKU may only be represented as a single location, investigation is done into the allocation of multiple non-adjacent locations to a single SKU. This subproblem is further discussed in Chapter 5.

3.2.3 PLAP

The largest of the subproblems which links all the other subproblems is the PLAP. Here a set of SKUs needs to be allocated to picking lines in such a way as to minimise the entire time taken to pick all the SKUs. An issue which filters down to the lower subproblems is that of allocating multiple locations to a single SKU on a picking line. This needs to be investigated and requires that solution methods for the OSP and SLP subproblems must be able handle this case. This subproblem is, however, left for further studies.

CHAPTER 4

Order sequencing problem

Contents

The order sequencing problem (OSP) may be described as the sequencing of all the orders in a wave, for pickers, for a given pickling line with fixed SKU positions such that the total picking time is minimized. The modelling and solving of the OSP will be discussed for the remainder of this chapter.

4.1 Subproblem description

When the actual picking starts on a picking line each picker is assigned an order. Pickers may also join or leave the picking line during a wave of picking. Once a picker is assigned an order, he must complete that order before moving on to another order. Orders are therefore sequentially picked by pickers. A picker starts an order by receiving instructions to proceed to the next required SKU. A picker may start an order at any location, however, physical picking will only begin at the first location which holds a required SKU. Pickers are only allowed to move in a clockwise direction around the picking line ensuring that locations are always passed in the same sequence and pickers are able to overtake one another while picking. Figure 4.1 illustrates a possible position for a picker to begin an order. Although the picker does not need to pick from location P the distance travelled to completely pick the order is calculated from location P. Thus from a distance travelled perspective the order is viewed as if it started at location P

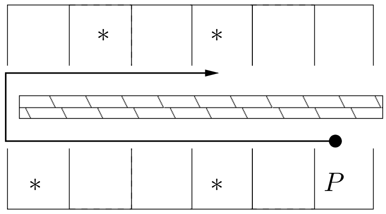


Figure 4.1: *A schematic representation of a possible starting location and distance travelled to complete an order. The current location of the picker when requesting a new order is indicated by a P. All required SKUs for the order are indicated with an asterisk.*

The following assumptions were made to model the problem:

1. It is assumed that stockouts do not occur during a wave of picking. In practice this is an infrequent event as sufficient stock may be stored at each location. The causes and implications of stockouts are therefore not considered.

2. All pickers can freely pass one another. In practice pickers will tend to only pass each other when one picker is picking and the other is walking to a further location.

3. The time taken to physically pick a SKU is constant over all the orders. In reality pick times are stochastic in nature. No focus is placed on improving the physical action of picking items in this thesis and it is assumed that the time required to pick items is fixed regardless of when and where it is picked. Furthermore, all the picks need to be made regardless of the sequence in which the orders are picked.

4. A picker walks at a constant speed. In reality this may not be the case, but over the long term a picker would show stable walking speeds. This assumption allows for a transformation from time to distance.

5. An order may start at any location regardless of whether the order requires the SKU at that location and the order will finish at the last location where a SKU is picked. This is illustrated in Figure 4.1.

6. A picker may not physically pick the first SKU of a new order from the same location as the last pick of the previous order. If the next order requires that same SKU the picker must either move to another location which holds that SKU or complete an entire cycle to pick at that location again. This assumption is due to a managerial decision to improve pick accuracy.

7. SKUs may be allocated multiple locations. If an order requires a SKU which is in multiple locations the first available location will be used.

4.2 Model

The objective of the model is to complete all orders in the shortest possible time. Following the above mentioned assumptions the only changeable time considered is the walking time. This in conjunction with assumption 4 which creates a correlation between time and distance allows for the model objective to be equivalent to minimising the distance travelled by all the pickers to complete all orders.

Due to the ability of pickers to pass each other a single sequence of orders may be cut or split in order to accommodate multiple pickers. Another option may be to initially assign order sets to pickers which are individually optimised. The model will therefore focus on creating a single sequence of orders for a single picker.

Based on the assumptions provided the OSP may be viewed as a variation of a unidirectional carousel system. Many studies have been done to sequence orders on a bidirectional carousel system and near optimal solutions have been developed. In a bidirectional carousel system the sequence in which individual SKUs in an order are picked, known as pick strategy is a non-trivial problem, where in the OSP case the nearest required SKU in a clockwise direction is always picked. Bartholdi & Platzman [3] showed that an optimal sequence for bidirectional carousel systems may be found in linear time. A number of heuristic solutions to this problem have also been investigated. Litvak & Adan [22] introduced the m-step method, where the carousel system changes direction when at most m SKUs have been picked.

Bartholdi & Platzman [3] further considered the problem of sequencing multiple orders on a bidirectional carousel system. A hierarchical heuristic was introduced by Bartholdi & Platzman [3] where orders are forced to be picked on their shortest spanning interval (SSI) and these SSIs were then linked up within 1 cycle. Van den Berg [32] introduced a similar approach where the SSIs are linked up by means of a rural postman problem approach.

Using SSIs in the bidirectional case yields optimal solutions, but it can easily be shown that using this methodology of picking all order on their SSIs in some cases yields worse case scenarios in terms of the distance traversed in the unidirectional case.

A number of definitions are needed to describe and model the OSP:

Definition 1. *The span of an order is the smallest set of locations passed to complete the order given a starting location.*

A starting position has a unique span associated with it, because an order must be completed once it is started. A span for order k may be represented by $S_k^i = \langle i, e_k^i \rangle$, where e_k^i is the closest ending location of order k starting at location i. Definition 1 implies that the last location of a span will hold a SKU required by the order.

Let the length of a span be denoted by $|S_k^i|$ and defined by the formula

$$|S_k^i| = |\langle i, e_k^i \rangle| = \begin{cases} m & \text{if } i = e_k^i \text{ and} \\ (e_k^i - i + m) \mod m & \text{otherwise,} \end{cases} \tag{4.1}$$

where m is the number of locations in the picking line.

Definition 2. *A minimum span for an order is a span of smallest size for that order.*

The minimum spans for order k are all spans S_k^i such that

$$|S_k^i| \leq |S_k^j| \qquad \forall j, \tag{4.2}$$

and we will denote the length of the minimum spans for order k as $|S_k^{\min}|$.

Each order may be assigned any one of the locations in the picking line as a starting position. However, this starting position needs to follow on from the preceding order and therefore the span on which an order is picked is dependent on the final location of the previous order. These dependencies on the previous order and also the final location allows for the OSP to be viewed as a variant of the travelling salesman problem where the distance between two orders are dependent on the complete sequence of orders which have been picked before.

Let the duple (i, k) represent order k starting at location i, implying the order is picked on S_k^i. Because the order can only be assigned one starting location, the problem may be viewed as a generalized travelling salesman problem (GTSP). Define the set $\mathcal{N}$ as the set of all duples (i, k). Let the sets $\mathcal{C}_1, \mathcal{C}_2, \ldots \mathcal{C}_n$ be a proper partition of the set $\mathcal{N}$, where $\mathcal{C}_k = \{(1, k), (2, k), \ldots, (n, k)\}$ and represents all the possible ways in which orders may be picked. The set $\mathcal{N}$ may be viewed as the vertices on a digraph with each edge representing the connection between orders with distinct starting locations and let D be the distance matrix associated with the digraph. Define $\hat{\mathcal{D}}$ to be the set of edges on the digraph. The objective of the GTSP is to find a cycle of edges $\mathcal{F} \subseteq \hat{\mathcal{D}}$ such that at least one vertex, or duple, in each set $\mathcal{C}_k$ is visited. A variant of the GTSP known as the equality generalized travelling salesman problem (E-GTSP), where only a single vertex is to be visited in each set $\mathcal{C}_k$ is considered here [11]. The goal of the E-GTSP may be viewed as choosing a vertex subset $\mathcal{V} \subseteq \mathcal{N}$, such that $|\mathcal{V} \bigcap \mathcal{C}_k| = 1$ for $k = 1, 2, 3, \ldots, n$. In order to formulate the E-GTSP based on a formulation by Fischetti *et al.* [11] the following entities need to be defined.
For each $\mathcal{V} \bigcap \mathcal{C}_i$ let

$\hat{\mathcal{D}}(\mathcal{V})$ be the set of edges where both vertices are contained in $\mathcal{V}$,

$\delta(\mathcal{V})$ be the set of edges between vertices contained in $\mathcal{V}$ and those not contained in $\mathcal{V}$,

$\mu(\mathcal{V}) = |\{h : \mathcal{C}_h \subseteq \mathcal{V}\}|$ and

$\eta(\mathcal{V}) = |\{h : \mathcal{C}_h \bigcap \mathcal{V} \neq \emptyset\}|$.

To formulate the E-GTSP as a mixed integer model, let

$$x_e = \begin{cases} 1 & \text{if edge } e \in \hat{\mathcal{D}} \text{ is traversed in the solution} \\ 0 & \text{otherwise,} \end{cases}$$

$$y_v = \begin{cases} 1 & \text{if vertex } v \in \mathcal{N} \text{ is visited in the solution} \\ 0 & \text{otherwise.} \end{cases}$$

and

$$d_e \text{ be the length of edge } x_e. \tag{4.3}$$

In terms of the above definitions the objective is to

$$\text{minimise} \quad \sum_{e \in \hat{\mathcal{D}}} d_e x_e \tag{4.4}$$

subject to

$$\sum_{e \in \delta(\mathcal{V})} x_e = 2y_v \qquad v \in \mathcal{N}, \tag{4.5}$$

$$\sum_{v \in \mathcal{C}_h} y_v = 1 \qquad h = 1, 2, 3, \ldots, n, \tag{4.6}$$

$$\sum_{e \in \delta(\mathcal{V})} x_e \geq 2(y_i + y_j - 1) \qquad \mathcal{V} \subset \mathcal{N}, 2 \leq |\mathcal{V}| \leq n-2, i \in \mathcal{V}, j \in \mathcal{N}/\mathcal{V}, \tag{4.7}$$

$$x_e \in 0,1 \qquad e \in \hat{\mathcal{D}}, \tag{4.8}$$

$$y_v \in 0,1 \qquad v \in \mathcal{N}. \tag{4.9}$$

The objective function (4.4) minimises the sum of the lengths of all the edges in the solution with constraint set (4.5) ensuring that every vertex that is visited has two edges incident with it. Constraint set (4.6) ensures that a vertex from each cluster is visited. The subtour breaking constraints are represented by constraint set (4.7).

Several additional, and different, exact formulations for the BOSP were also investigated. One of these formulations is given and discussed in this section with the remainder discussed in Appendix A. The first formulation is a more detailed formulation based on the E-GTSP model given in Fischetti *et al.* [11]. In order to model the problem in this way let

$$x_{ikl} = \begin{cases} 1 & \text{if order } k \text{ starting at location } i \text{ is followed by order } l \\ 0 & \text{otherwise} \end{cases}$$

and

p_k be the position of order k within the order sequence.

The following parameters are set in the model. Let

n be the total number of orders,

m be the total number of locations,

$|S_k^i|$ be the length of the span for order k starting at location i and

$$e_{ikj} = \begin{cases} 1 & \text{if order } k \text{ starting at location } i \text{ is completed at location } j \\ 0 & \text{otherwise.} \end{cases}$$

The objective is then to

$$\text{minimise} \quad \sum_{i=1}^{m} \sum_{k=1}^{n} \sum_{l=1}^{n} |S_k^i| x_{ikl} \tag{4.10}$$

subject to

$$\sum_{i=1}^{m} \sum_{k=1}^{n} x_{ikl} = 1 \qquad l = 1, \ldots, n, \tag{4.11}$$

$$\sum_{i=1}^{m} \sum_{l=1}^{n} x_{ikl} = 1 \qquad k = 1, \ldots, n, \tag{4.12}$$

$$\sum_{l=1}^{n} x_{m1l} = 1 \tag{4.13}$$

$$p_1 = 1, \tag{4.14}$$

$$p_k - p_l + n\sum_{i=1}^{m} x_{ikl} \leq n-1 \qquad k=1,\ldots,n, \quad l=2,\ldots,n, \tag{4.15}$$

$$\sum_{l=1}^{n} x_{ikl} - \sum_{p=1}^{m}\sum_{q=1}^{n} x_{pqk}e_{pqj} \leq 0 \qquad i=1,\ldots,m, \quad k=1,\ldots,n, \tag{4.16}$$

$$x_{ikl} \in \{0,1\} \qquad i=1,\ldots,m, \quad k=1,\ldots,n, \quad l=1,\ldots,n, \tag{4.17}$$

$$p_k \geq 0 \qquad k=1,\ldots,n. \tag{4.18}$$

The objective function (4.10) minimises the total distance travelled by a picker in terms of locations passed. Constraint sets (4.11) and (4.12) ensure that each order is only completed once. Constraint sets (4.13) and (4.14) ensure that the first order (which is a dummy order) is completed first and that it starts at location 1. This accounts for pickers entering the system from the first location and allows for the first order to pick from location 1. Constraint set (4.15) ensures that no subtours are generated. This constraint set is based on MTZ constraints discussed in Punnen [27]. Constraint set (4.16) ensures that the starting point of the next order in the sequence follows on the ending point of the previous order. The dimensions of this formulation are n^2m+n variables (of which n^2m are binary) and $n^2+2n+nm$ constraints. For a standard size instance faced by Pep the number of variables are in excess of 8×10^7 and the number of constraints in excess of 1.5×10^6.

4.3 Lower bounds

The exact formulation was found to be too large to solve. Heuristic and metaheuristic methods are therefore needed to solve the OSP. To measure the effectiveness of any heuristic or metaheuristic a good lower bound is necessary.

Typical TSP lower bounds include removing subtour breaking constraints, removing subtour generation constraints linearisation of variables. If subtour generation is removed it is easy to show that all orders will be picked on minimum spans, (or SSIs in terms of bidirectional carousels). In testing not even the continuous LP relaxations to the exact formulations could be solved, using Lingo 11 [21]. Memory issues arose due to the large data sets required typically in excess of 8×10^7 variables and 1.5×10^6 constraints. A lower bound to the solution could therefore not be found using exact formulations and alternative methods to calculate lower bounds were considered.

An initial method was to sum the lengths of the minimum spans of all the orders giving a lower bound in terms of the total number of locations passed. This method does not take into account actual sequencing and is a very weak lower bound.

A further approach, which will be referred to as the maximal cut approach, uses an IP model to specifically generate a lower bound for the OSP. The approach attempts to use the cyclical structure of the picking line by considering the number of full cycles traversed rather than the number of locations passed. To model the OSP in terms of cycles traversed the following definition is made.

Definition 3. *The cut of a location is the number of spans passing that location.*

The cut for each location forms a lower bound for the number of cycles needed to pick a set of spans as it represents a minimum number of times a location must be passed to pick all of the spans. A lower bound (in terms of cycles) for the OSP may be determined by assigning starting positions to all orders while minimising the largest (or maximal) cut(s). To strengthen the lower bound these starting positions are then paired up with ending positions forcing all orders to follow directly after a unique preceding order, but the formulation does not break possible subtours.

The following model achieves this objective. Let

$$x_{ik} = \begin{cases} 1 & \text{if order } k \text{ starts at location } i \\ 0 & \text{otherwise,} \end{cases}$$

and

C be the maximal cut.

The following parameters are set in the model. Let

n be the total number of orders,

m be the total number of locations,

$$\bar{d}_{ikj} = \begin{cases} 1 & \text{if order } k \text{ starting at location } i \text{ passes location } j \\ 0 & \text{otherwise} \end{cases}$$

and

$$e_{ikj} = \begin{cases} 1 & \text{if order } k \text{ starting at location } i \text{ is completed at location } j \\ 0 & \text{otherwise.} \end{cases}$$

In terms of these symbols the objective is to

$$\text{minimise} \quad C \tag{4.19}$$

subject to

$$\sum_{i=1}^{m} x_{ik} = 1 \qquad k = 1, \ldots, n, \tag{4.20}$$

$$\sum_{i=1}^{m} \sum_{k=1}^{n} \bar{d}_{ikj} x_{ik} \leq C \qquad j = 1, \ldots, m, \tag{4.21}$$

$$\sum_{k=1}^{n} x_{j+1,k} - \sum_{i=1}^{m} \sum_{k=1}^{n} x_{ik} e_{ikj} = 0 \qquad j = 1, \ldots, m-1, \tag{4.22}$$

$$\sum_{k=1}^{n} x_{1k} - \sum_{i=1}^{m} \sum_{k=1}^{n} x_{ik} e_{ikm} = 0, \tag{4.23}$$

$$x_{ik} \in \{0, 1\} \qquad i = 1, \ldots, m, \quad k = 1, \ldots, n. \tag{4.24}$$

The objective function (4.19) minimises the maximal cut. Constraint set (4.20) ensures that each order is allocated one starting position. Constraint set (4.21) calculates the size of the maximal cut. Pairing constraint set (4.22) and constraint (4.23) ensures that the starting position of every order may be paired with an ending position of another order. These constraints do not break subtours or link up orders but are additional feasibility constraints as a feasible solution requires each order to directly follow on another order. The dimensions of this formulation are $nm + n$ variables and $n + 3m$ constraints. A typical real life instance thus yields a number of variables in excess of 67200 and a number of constraints in excess of 1368, which is considerably smaller than formulation (4.10)–(4.18).

The parameter $\bar{d}_{ikj}$ is generated by assigning a value of 1 to every location j in the span of order k starting at location i. The pairing of starting and ending positions is done by pairing ending positions with starting positions occurring within one location of each other.

As an example consider the picking line with the following configuration P_1-P_2-P_3-P_1-P_4-P_5 and the order k which requires the following SKUs P_1, P_2, P_4. A schematic representation of this layout is given in Figure 4.2. If order k starts at location 1 then $\bar{d}_{1k1} = \bar{d}_{1k2} = \bar{d}_{1k3} = \bar{d}_{1k4} = \bar{d}_{1k5} = 1$ and $\bar{d}_{1k6} = 0$ must be generated, or if order k starts at location 5 then $\bar{d}_{5k1} = \bar{d}_{5k2} = \bar{d}_{5k5} = \bar{d}_{5k6} = 1$ and $\bar{d}_{5k3} = \bar{d}_{5k4} = 0$ must be generated.

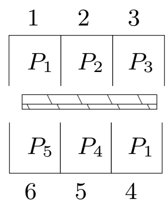


Figure 4.2: *A schematic representation of the layout of an example picking line with 6 locations and 5 SKUs.*

The solution for the maximal cut model may be represented as two sets and an objective function value. Let $\mathcal{S}$ represent the starting positions, $\mathcal{E}$ represent the ending positions and C the maximum of all cuts. The solution given may not be optimal as the starting and ending positions of all the orders may not allow for the generation of a single complete tour. A procedure was developed to create a feasible solution for the OSP using $\mathcal{S}$ and $\mathcal{E}$ which minimises the number of additional locations which need to be passed. This procedure requires Algorithm 1 which generates the minimum number of subtours within the solution. These subtours are then linked up and the resulting solution to the OSP is shown, by means of Theorem 1 and Theorem 2, to be at most 1 cycle greater than C.

Theorem 1 states that Algorithm 1 will find a set $\mathcal{T}$ of subtours containing all orders such that $|\mathcal{T}|$ is at most the number of locations. Theorem 2 states that a feasible solution may be constructed from these $|\mathcal{T}|$ subtours by simply linking the starting and ending points of each subtour such that the link up cost is at most 1 cycle.

Theorem 1. *Algorithm 1 produces a set of closed subtours $\mathcal{T}$ such that $|\mathcal{T}| \leq m$.*

Proof. It must be shown that all subtours in the set $\mathcal{T}$ will close and that $|\mathcal{T}| \leq m$. Assume that Algorithm 1 returns an open subtour. This indicates that the ending position of this open subtour has no corresponding starting position which would indicate that $\mathcal{S}$ and $\mathcal{E}$ do not fulfil pairing constraint set (4.22) and constraint (4.23). This contradicts the assumption that $\mathcal{S}$ and $\mathcal{E}$ was generated by means of formulation (4.20) to (4.24).

Algorithm 1: Subtour generation heuristic

Input: A set $\mathcal{S}$ of starting positions and a set $\mathcal{E}$ of ending positions for the spans obtained from formulation (4.20) to (4.24).
Output: A set $\mathcal{T}$ of subtours that links up all the orders

while *All orders have not been allocated to a subtour* **do**
  Generate a new subtour t_i with the first available unallocated order;
  Let the current ending position of t_i be location j;
  while t_i *is not closed* **do**
    if *An unallocated order exists which has a starting position corresponding to* $j + 1 \mod m$*;*
    then
      Add this order to the end of the open subtour;
    end
    else
      Close the subtour by connecting the last order to the first order;
    end
  end
end

Assume that $|\mathcal{T}| > m$. There is at most m possible starting and ending positions for subtours. According to the pigeon hole principle there must exists at least two subtours u and v in $\mathcal{T}$ which start at the same location i. Assume that subtour u was generated before subtour v. This indicates that at some point in the algorithm subtour u ended at location i, and closed, where it could have selected the first order in subtour v as it was at that stage an unallocated order. This contradicts Algorithm 1 which closes a subtour only once all unallocated orders with a feasible starting position have been allocated to a subtour.

□

This set of subtours $\mathcal{T}$ can be linked up to make a single subtour which is at most 1 cycle longer than the sum of the lengths of the individual subtours in $\mathcal{T}$. This results is proved in Theorem 2

Theorem 2. *A set of subtours $\mathcal{T}$ generated by means of Algorithm 1 may be connected to form a single tour of length at most $C + 1$ cycles.*

Proof. Order the closed subtours in $\mathcal{T}$ by increasing value of their starting positions. Connect these subtours by joining the ending and starting positions sequentially using this ordered set. This connection cost of linking the subtours is at most 1 cycle as all locations may be visited in at most 1 cycle. The number of cycles required to complete this single tour is the sum of the number of cycles required by each individual subtour (C) and the number of cycles required to link all the subtours (1). Therefore a set of subtours $\mathcal{T}$ generated by means of Algorithm 1 may be connected to form a single tour of length at most $C + 1$ cycles.

□

4.4 Implementing solutions in the Pep context

It has been assumed for modelling purposes that only one picker operates in a picking line. However, this is not the case and any practical implementation needs to consider multiple pickers. The robustness of the maximal cut approach with regards to multiple pickers needs to be investigated for use in Pep.

Algorithm 1 generates a set of static subtours which must be completed. These subtours cannot simply be allocated to different pickers as their lengths are not necessarily equal and work would not be evenly distributed between pickers. A procedure was developed based on Algorithm 1 to pass orders to multiple pickers in real time. This procedure is summarised in Algorithm 2. The procedure attempts to pass orders with starting positions matching the current position of the picker and if no match can be found a closest order is then passed to the picker.

Algorithm 2: Dynamic allocation of orders to pickers

```
Data: A set of starting ending positions for the orders S and E
while All orders have not been completed do
    if A picker requests an order from location l then
        if An unallocated order exists which has a starting position
        corresponding to l;
        then
            Pass this order to the picker;
        end
        else
            Pass a closest uncompleted order to the picker;
        end
    end
end
```

To compare the dynamic allocation of orders to the maximal cut model a discrete event simulation model was developed. Pickers were assigned picking and walking speeds based on real life data and were assumed to pass each other freely. The time required to complete an order was correlated to the picker's speed and the SKUs required by the order. The picking process was simulated by assigning new orders whenever a picker completed his current order using Algorithm 2. The faster pickers are thus expected to complete more orders than the slower pickers. The model calculated the number of cycles traversed collectively by all pickers and was used to calculate the additional cycles traversed when multiple pickers are used.

To test the simulation historical data sets were obtained from Pep. The historical data sets differed in terms of number of SKUs and number of orders. Due to the current managerial constraints there are no historical data sets for cases where SKUs are duplicated on a picking line. Instances of this type were generated by duplicating the 10 most dense SKUs for each historical picking line. These generated instances still had the same order sets associated with the historical data but now had additional bay locations for the duplicated SKUs.

The simulation was run 20 times for each data set with each run associated with different picker speeds. Testing using this simulation model and historical data from Pep was done using a Intel(R) Core(TM)2 Duo 3GHz with 3.7 GB ram running UBUNTU 9.10 [31] with all programming done in JAVA [30]. A summary of these results is given in Table 4.1.

The results presented in Table 4.1 suggest that the dynamic allocation of orders to pickers does not significantly influence the total walking distance of the pickers. The robustness of the maximal cut approach for the stochastic case of the OSP with multiple pickers appears viable and will further be used as the standard measure of solution quality for any OSP instance.

The maximal cut approach was further compared to the current approach used by Pep. Currently the VRS uses a fixed list system for allocating orders to pickers. Whenever a picker requests a new order the VRS will allocate the next order in this fixed list regardless of the pickers current position. Taking the stochastic nature of picking into account this fixed list system may be described as a random allocation procedure for the maximal cut approach as the sequence of

Data set	Size (n, SKUs, m)	Additional cycles μ	σ	Data set	Size (n, SKUs, m)	Additional cycles μ	σ
A	(1262, 49, 49)	0.45	0.45	A′	(1262, 49, 59)	5.8	0.76
B	(1264, 54, 54)	0	0	B′	(1264, 54, 64)	3.15	0.93
C	(1265, 51, 51)	1	0.4	C′	(1265, 51, 61)	3.55	0.95
D	(1263, 56, 56)	9.35	1.03	D′	(1263, 56, 66)	4.35	1.43
E	(1264, 51, 51)	7.6	2.04	E′	(1264, 51, 61)	0.45	0.25
F	(1258, 53, 53)	4.75	1.59	F′	(1258, 53, 63)	11.25	1.29
G	(1260, 56, 56)	2.7	1.61	G′	(1260, 56, 66)	0.45	0.35
H	(1244, 54, 54)	1.25	1.19	H′	(1244, 54, 64)	2.5	0.75
I	(1264, 56, 56)	0.35	0.23	I′	(1264, 56, 66)	0.3	0.31
J	(1258, 55, 55)	2.15	1.53	J′	(1258, 55, 65)	5.6	0.34
K	(943, 63, 63)	0.85	0.43	K′	(943, 63, 73)	0.7	0.61
L	(846, 56, 56)	0.45	0.35	L′	(846, 56, 66)	3.85	0.83
M	(728, 51, 51)	0.95	1.05	M′	(728, 51, 61)	0.5	0.55
N	(396, 63, 63)	10.5	1.65	N′	(396, 63, 73)	3.8	1.56
O	(733, 55, 55)	2.8	0.76	O′	(733, 55, 65)	1.3	1.11
P	(242, 64, 64)	4.1	0.99	P′	(242, 64, 74)	1.3	0.51
Q	(574, 48, 48)	3.2	0.46	Q′	(574, 48, 58)	3.1	0.69
R	(90, 48, 48)	0.35	0.23	R′	(90, 48, 58)	1	0
S	(158, 55, 55)	3.75	0.29	S′	(158, 55, 65)	0.85	0.63
T	(82, 51, 51)	1.55	0.45	T′	(82, 51, 61)	0.4	0.24
U	(80, 56, 56)	0	0	U′	(80, 56, 66)	0.3	0.21
V	(89, 42, 42)	1.2	0.16	V′	(89, 42, 52)	0.8	0.16

Table 4.1: *The number of additional cycles walked by a set of 8 pickers for different OSP instances when using Algorithm 2 to dynamically allocate orders. Each picker was assigned a random walking and picking speed all of which were within 25% of each other. The average number of additional cycles traversed collectively for 8 pickers over 20 simulation runs is indicated by μ and the standard deviation by σ.*

orders which an individual picker picks is influenced by the pick speeds of all the other pickers. The maximal cut approach was therefore compared to the historical results from Pep and the results presented in Table 4.2. Figure 4.3 illustrates the percentage improvement of the maximal cut approach compared to Pep's approach.

The results in Table 4.2 suggest significant improvements on the current method used by Pep. In some cases the number of cycles traversed is halved in comparison to historical data.

To tests the hypothesis that the current sequencing approach used by Pep may be described as a random approach a random sequencing approach was tested and the results compared to the historical results. The historical results were gathered by calculating the number of cycles traversed by each picker used during the wave of picking. Because of the number of pickers as well as the effects of lunch breaks and shift changes pickers do not complete full cycles during their shifts. These semi completed cycles were rounded up in the calculation which explains why the historical results are significantly higher than the random approach which sequences orders for a single picker. The results do, however, suggest that a random approach is a prudent estimator for the current order sequencing approach as in many cases the random approach performs better than the historical results.

For cases where duplicated SKUs are present no historical results are available as duplicating SKUs on the picking line is currently not allowed due to system constraints. A random sequencing approach will therefore be used to estimate Pep's current approach. The results for the duplicated instances are presented in Table 4.3.

The results presented in Table 4.3 indicate that the maximal cut approach would still outperform

Data set	Size (n, SKUs, m)	Maximal cut	Historical results	Random approach μ	σ
A	(1262, 49, 49)	1232	1262	1235.5	1.4
B	(1264, 54, 54)	1226	1255	1234.6	1.8
C	(1265, 51, 51)	1161	1254	1220.3	2.1
D	(1263, 56, 56)	1072	1224	1202.1	2.6
E	(1264, 51, 51)	1069	1234	1209.2	3
F	(1258, 53, 53)	1005	1222	1190.5	3.3
G	(1260, 56, 56)	955	1227	1168.7	2.2
H	(1244, 54, 54)	992	1242	1174.4	2.4
I	(1264, 56, 56)	947	1202	1168.3	3.4
J	(1258, 55, 55)	1025	1177	1165.7	2.6
K	(943, 63, 63)	259	640	627.3	7.4
L	(846, 56, 56)	232	615	549.6	6.2
M	(728, 51, 51)	152	457	445.4	6.6
N	(396, 63, 63)	90	224	248.5	4.3
O	(733, 55, 55)	125	461	432.3	4.1
P	(242, 64, 64)	45	142	145.4	4.7
Q	(574, 48, 48)	80	324	326.2	7.5
R	(90, 48, 48)	7	40	46.5	2.8
S	(158, 55, 55)	14	82	83.7	3.5
T	(82, 51, 51)	8	36	43	1.9
U	(80, 56, 56)	6	38	42.1	2.7
V	(89, 42, 42)	9	40	47.8	3.1

Table 4.2: *A comparison between the number of cycles traversed for the maximal cut approach, Pep's historical results and a random approach representing Pep's current order sequencing method. No duplicated SKUs are present for any of the data sets. The average and standard deviation over 20 runs for the random approach is given as μ and σ respectively.*

the approach used by Pep for both types of instances (with and without duplicated SKUs).

Considering the duplication of the dense SKUs it may be seen that savings in terms of cycles are realised for both the maximal cut and random approaches although the savings are more prominent when using the maximal cut. It may be noted that for data set K the addition of duplicates increases the number of cycles traversed. This is not caused by the inclusion of duplicated SKUs but rather different SKU locations for the two data sets.

For the case where duplicated SKUs are present it should be noted that the spans on which orders are picked is known. Therefore if a SKU is allocated multiple locations the orders which pick that SKU would have each been assigned one of the multiple locations. This allows for the accurate calculation of the required stock at each of the locations. Therefore the issue of stockouts when duplicated SKUs are present is eliminated.

4.5 Heuristic and metaheuristic approaches

Although the OSP may be solved within one cycle of optimality by means of the maximal cut approach in a feasible amount of time (3 minutes) for a once of calculation, the computational time is too excessive for use in the SLP. Heuristic and metaheuristic methods were therefore developed in order to solve the OSP in shorter times. The algorithms do not necessarily need to directly produce a feasible solution to the OSP but may bound the value of the maximal cut by using infeasible solutions in order to quickly compare solution quality between different SKU allocations. All the approaches presented here solve the problem for a single picker as the

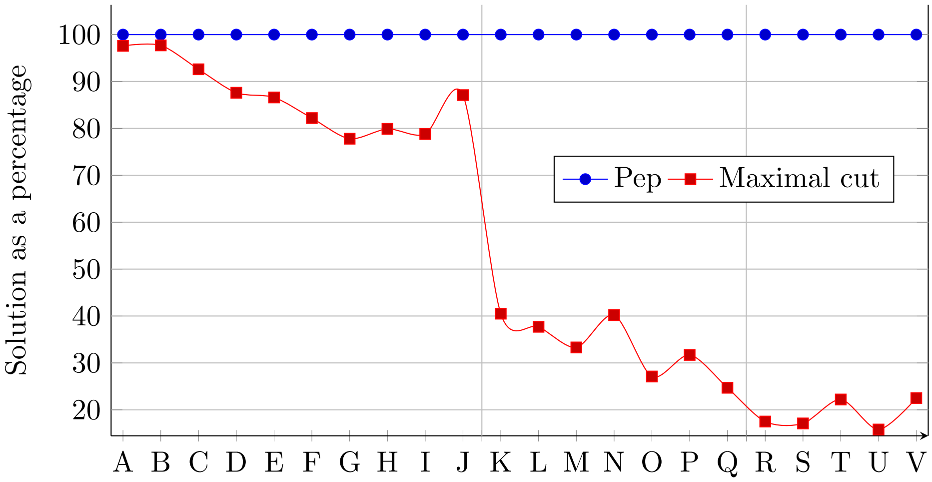


Figure 4.3: *A graphical illustration of the percentage improvement in the number of cycles traversed of the maximal cut approach in comparison to Pep's approach expressed as a percentage of Pep's approach.*

Data set	Size (n, SKUs, m)	Maximal cut	Historical results	Random approach μ	σ
A′	(1262, 49, 59)	925	–	1145.7	3.4
B′	(1264, 54, 64)	931	–	1151.7	2.9
C′	(1265, 51, 61)	878	–	1118.7	2.1
D′	(1263, 56, 66)	911	–	1142.3	4
E′	(1264, 51, 61)	890	–	1123	3
F′	(1258, 53, 63)	895	–	1123.8	4.2
G′	(1260, 56, 66)	900	–	1128.9	3.3
H′	(1244, 54, 64)	945	–	1147.7	2.6
I′	(1264, 56, 66)	898	–	1132.4	2.8
J′	(1258, 55, 65)	778	–	1064.6	4.1
K′	(943, 63, 73)	269	–	596.5	6.4
L′	(846, 56, 66)	194	–	482.2	5.2
M′	(728, 51, 61)	114	–	366.7	5.9
N′	(396, 63, 73)	69	–	211.3	3
O′	(733, 55, 65)	104	–	380.8	5
P′	(242, 64, 74)	55	–	141.9	5
Q′	(574, 48, 58)	52	–	251.5	3.8
R′	(90, 48, 58)	4	–	38.5	2.5
S′	(158, 55, 65)	13	–	73.4	2.2
T′	(82, 51, 61)	6	–	36.7	3
U′	(80, 56, 66)	5	–	35.1	2.2
V′	(89, 42, 52)	6	–	38.3	2.6

Table 4.3: *A comparison between the number of cycles traversed for the maximal cut approach, Pep's historical results and a random approach representing Pep's current order sequencing method. Duplicated SKUs are present for all of the data sets. The average and standard deviation over 20 runs for the random approach is given as μ and σ respectively.*

maximal cut formulation may be solved for multiple pickers at a later stage once the SLP has been solved.

4.5.1 Greedy approaches

A greedy heuristic was developed to solve the OSP where the closest order is chosen as the next order at each iteration. Four variations of a greedy algorithm, namely G1–G4, were tested. All four variations use the same general structure summarised in Algorithm 3. Each variation, however, made use of a different measure to determine the closest order.

Algorithm 3: Greedy heuristic

```
Data: A set of orders and SKU locations
A current location L_c = 1
for The number of orders do
    Search for the closest order to L_c from all remaining orders;
    Add the closest order to the sequence;
    Set L_c to the ending position of the added order;
end
```

In variation G1 when at location i the closest order would be seen as order k where

$$k = \arg\min_{l} |S_l^i|. \tag{4.25}$$

The closest order may be seen as the order which may be completed within the smallest number of locations from the current location.

Consider the example given in Figure 4.4. Using variation G1 the closest order would be order C. It may, however, be more advantageous to pick order A as one would then be picking on a minimum span for order A and leave order C for a later selection. This lead to the introduction of variation G2.

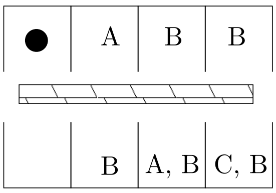


Figure 4.4: *A schematic representation of an example picking line with 8 locations and subset of 3 orders (A, B, C). The letters indicate if an order needs a SKU at that location. The black circle indicates the current position of the picker.*

The closest order in variation G2 is determined by comparing a relative completion length. This is done by choosing the next order k by means of

$$k = \arg\min_{l} \frac{|S_l^i|}{|S_l^{\min}|} \tag{4.26}$$

where $|S_k^{\min}|$ is the length of the minimum span of order k. If $\frac{|S_k^i|}{|S_k^{\min}|}$ equals 1 we know that the length of order k is at a minimum. This variation gives preference to orders that are picked on their minimum spans.

Consider again the example in Figure 4.4. Although variation G2 now picks on the minimum span of order A it should be noted that two minimum spans exists for order A. It may be more advantageous to pick order B as the number of picks per distance traversed for order B is greater than A. This idea of valuing the shorter spans of orders with many SKUs is introduced with variation G3. Variation G3 uses the relative measure given by

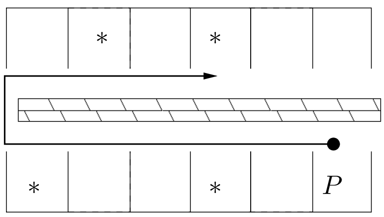
$$k = \arg\min_{l} \frac{|S_l^i|}{\lfloor S_l^{\min} \rfloor} \tag{4.27}$$

to determine the next order k, where $\lfloor S_k^{\min} \rfloor$ is the number of SKUs in order k. This variation considers the number of possible starting positions of each order. The smaller the number of starting positions the greater preference is given to start the order on its minimum span.

The final variation, G4, uses a combined relative measure of both variations G2 and G3 such that the next order k is determined by

$$k = \arg\min_{l} \frac{|S_l^i|}{|S_l^{\min}| \cdot \lfloor S_l^{\min} \rfloor} \tag{4.28}$$

where $\lfloor S_k^{\min} \rfloor$ is the number of SKUs in order k. This variation considers both the minimum span as well as the number of possible starting positions of each order. Figure 4.5 illustrates the calculation of these different measures with the aid of an example.



Parameters			Distance measure	
$\lvert S_k^i \rvert$	=	9	G1 =	9
$\lvert S_k^{\min} \rvert$	=	7	G2 =	9/7
$\lfloor S_k^{\min} \rfloor$	=	4	G3 =	9/4
			G4 =	9/28

Figure 4.5: *An example of the calculation of the different greedy distance measures. The figure on the left illustrates the required SKUs for an order indicated with an asterisk. P indicates the current position of the picker. the table on the right illustrates the different parameter values and distance measures for the order given the current position of the picker.*

All four of the variations of Algorithm 3 were tested using the same data sets as in §4.4. Testing was done using a Intel(R) Core(TM)2 Duo 3GHz with 3.7 GB ram running UBUNTU 9.10 [31]. The summary of the results for these algorithms is given in Tables 4.4 and 4.5.

The results in Table 4.4 (which are also graphically summarised in Figure 4.6) indicate that variation G1 of Algorithm 3 has the worst performance. Variation G2 shows the best performance for the 4 largest data sets (A-E) in terms of maximal cut with variation G4 in close second. For the smaller sized cases (F-J) variation G4 shows the best performance followed closely by both variation G2 and variation G3. All 4 variations (G1–G4) show equal performances for the smaller data sets (K-V). When considering percentage loss when using the greedy approach described in Algorithm 3 only data sets with maximal cuts greater than 200 (A-L) were considered as the relative measure would be more comparable. All variations of Algorithm 3 are on average outperformed by the maximal cut formulation by 5% for these data sets.

The data presented in Table 4.5 are graphically summarised in Figure 4.7. These results indicate that for OSP instances where duplicated SKUs are present variations G3 and G4 of Algorithm 3 perform the best. Variation G1 still shows the worst performance for all data sets. For all large data sets ($\mathrm{A}' - \mathrm{J}'$) variation G4 has the best solution and for the smaller data sets ($\mathrm{K}' - \mathrm{V}'$) variation G3 has the best solution followed closely by variation G4. The performance of variation G2 relative to the other variations has been significantly hampered by the inclusion of duplicated SKUs. This is evident by variation G2 not outperforming either variation G3 or variation G4 for any data set. The average percentage increase in cycles traversed by all variations has increased

Data set	Size (n, SKUs, m)	Greedy Algorithms G1	G2	G3	G4	Maximal cut
A	(1262, 49, 49)	1248	1247*	1252	1252	1232
B	(1264, 54, 54)	1243	1232*	1241	1241	1226
C	(1265, 51, 51)	1230	1180*	1184	1185	1161
D	(1263, 56, 56)	1202	1108*	1144	1133	1072
E	(1264, 51, 51)	1186	1123	1119	1111*	1069
F	(1258, 53, 53)	1196	1062	1055	1049*	1005
G	(1260, 56, 56)	1122	1014	1017	1013*	955
H	(1244, 54, 54)	1128	1031	1042	1030*	992
I	(1264, 56, 56)	1088	993	978	977*	947
J	(1258, 55, 55)	1199	1104	1068	1067*	1025
K	(943, 63, 63)	393	295	282*	290	259
L	(846, 56, 56)	305	245	241*	245	232
M	(728, 51, 51)	225	206	187*	189	152
N	(396, 63, 63)	184	138*	140	140	90
O	(733, 55, 55)	187	157	140*	150	125
P	(242, 64, 64)	90	65	58*	58*	45
Q	(574, 48, 48)	125	115	103*	109	80
R	(90, 48, 48)	10	9*	9*	9*	7
S	(158, 55, 55)	24	21*	22	22	14
T	(82, 51, 51)	11*	11*	11*	12	8
U	(80, 56, 56)	7*	8	8	8	6
V	(89, 42, 42)	14	13	11*	11*	9

Table 4.4: *A comparison of the number of cycles traversed between the different variations (G1–G4) of Algorithm 3 and the maximal cut for historical OSP instances where no duplicated SKUs are present. The best solution (Least number of cycles) is indicated by an asterisk. An upper bound for each instance may be seen as the number of orders n.*

to 7% with the inclusion of duplicated SKUs. This suggests that greedy approaches are less effective when duplicated SKUs are present. Based on these results variation G4 will be the variation used for the remainder of the study as it shows the most consistent performance for data with and without duplicated SKUs.

4.5.2 Relaxed maximal cut

Investigation was placed into reducing the complexity of the maximal cut formulation while still maintaining good bounds and it was discovered that if pairing constraint set (4.22) and constraint (4.23) were removed from formulation (4.20) to (4.24) the objective function value remained unchanged. In order two prove this the following definition is made.

Definition 4. *Adjacent maximal cuts are two maximal cuts such that one of the sets of locations separating them have no maximal cuts.*

Let $[i \circlearrowright j]$ represent a range of locations on a picking line starting from, and including, location i and moving in a clockwise direction to, and including, j. Round brackets excludes the corresponding starting and/or ending point of the range, for example $[i+1 \circlearrowright j] \equiv (i \circlearrowright j]$. Definition 4 implies that if cuts c_i and c_j are adjacent maximal cuts then the following must hold, all cuts c_w with $w \in (i \circlearrowright j)$ are not maximal.

Definition 4 implies that if cuts c_i and c_j, $i \leq j$, are adjacent maximal cuts then the following must hold, all cuts $c_{i+1}, c_{i+2}, \ldots, c_{j-1}$ are not maximal and/or all cuts $c_{j+1}, c_{j+2} \ldots, c_m, c_1, \ldots c_{i-1}$ are not maximal.

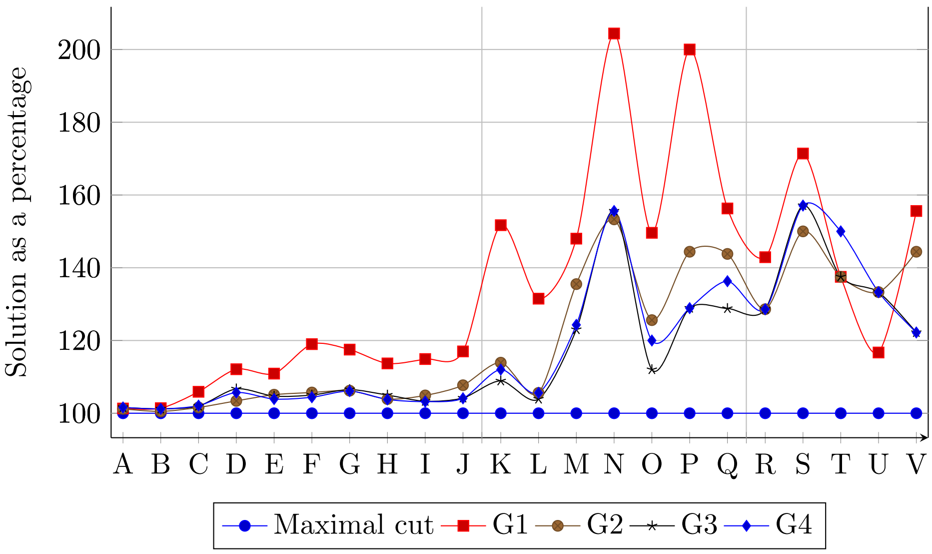


Figure 4.6: *A graphical illustration of the comparison between the different greedy variations (G1–G4) of Algorithm 3 and the maximal cut approach for historical data sets. The maximal cut approach represents the 100% benchmark.*

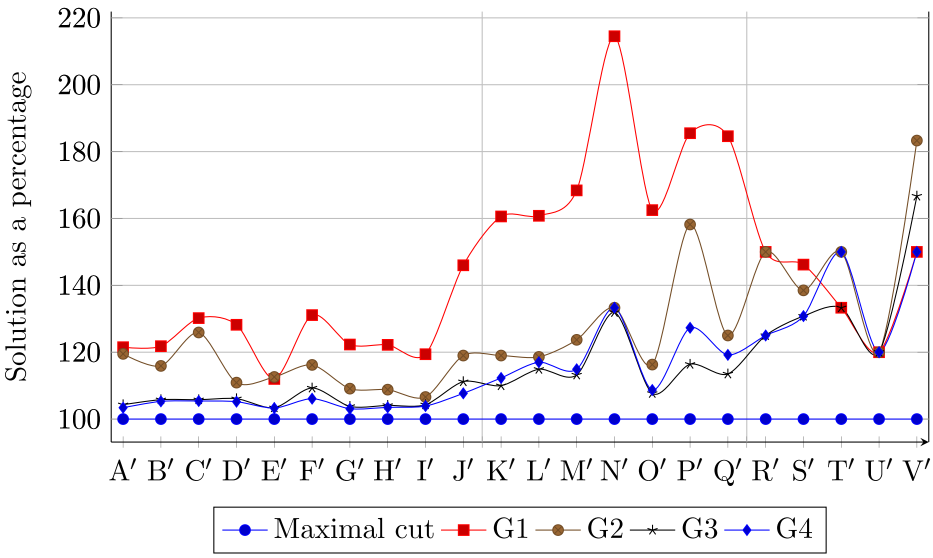


Figure 4.7: *A graphical illustration of the comparison between the different greedy variations (G1–G4) of Algorithm 3 and the maximal cut approach for OSP instances with duplicated SKUs. The maximal cut approach represents the 100% benchmark.*

Data set	Size (n, SKUs, m)	G1	G2	G3	G4	Maximal cut
A′	(1262, 49, 59)	1124	1105	965	956*	925
B′	(1264, 54, 64)	1134	1079	985	980*	931
C′	(1265, 51, 61)	1143	1105	929	925*	878
D′	(1263, 56, 66)	1168	1010	967	958*	911
E′	(1264, 51, 61)	997	1002	920	919*	890
F′	(1258, 53, 63)	1173	1040	978	950*	895
G′	(1260, 56, 66)	1101	982	934	928*	900
H′	(1244, 54, 64)	1155	1028	984	978*	945
I′	(1264, 56, 66)	1072	957	937	933*	898
J′	(1258, 55, 65)	1136	926	865	838*	778
K′	(943, 63, 73)	432	320	296*	302	269
L′	(846, 56, 66)	312	230	223*	227	194
M′	(728, 51, 61)	192	141	129*	131	114
N′	(396, 63, 73)	148	92	91*	92	69
O′	(733, 55, 65)	169	121	112*	113	104
P′	(242, 64, 74)	102	87	64*	70	55
Q′	(574, 48, 58)	96	65	59*	62	52
R′	(90, 48, 58)	6	6	5*	5*	4
S′	(158, 55, 65)	19	18	17*	17*	13
T′	(82, 51, 61)	8*	9	8*	9	6
U′	(80, 56, 66)	6*	6*	6*	6*	5
V′	(89, 42, 52)	9*	11	10	9*	6

Table 4.5: *A comparison of the number of cycles traversed between the different variations (G1–G4) of Algorithm 3 and the maximal cut for OSP instances where duplicated SKUs are present. The best solution (Least number of cycles) is indicated by an asterisk. An upper bound for each instance may be seen as the number of orders n.*

Let us consider the following relaxed version of formulation (4.20) to (4.24) with $\bar{C}$ representing a maximal cut. The objective is to

$$\text{minimise} \quad \bar{C} \tag{4.29}$$
subject to

$$\sum_{i=1}^{m} x_{ik} = 1 \qquad k = 1, \ldots, n, \tag{4.30}$$

$$\sum_{i=1}^{m}\sum_{k=1}^{n} \bar{d}_{ikj} x_{ik} \leq \bar{C} \qquad j = 1, \ldots, m, \tag{4.31}$$

$$x_{ik} \in \{0, 1\} \qquad i = 1, \ldots, m, \quad k = 1, \ldots, n. \tag{4.32}$$

The set of ending positions $\hat{\mathcal{E}}$ and the set of starting positions $\hat{\mathcal{S}}$ generated by formulation (4.29)–(4.32) may be connected to generate subtours without increasing the value of $\bar{C}$ then it is shown that constraint set (4.22) and constraint (4.23) do not increase the maximal cut.

To arrive at this result it must first be shown that for any solution of formulation (4.29)–(4.32) there will be an equal number of starting and ending positions between two adjacent maximal cuts. This is achieved by means of Theorem 3.

Theorem 3. *In a solution to formulation* (4.29)–(4.32) *the number of starting points between any two maximal cuts c_i and c_j is equal to the number of ending points between c_i and c_j.*

Proof. Let $[i, j]$ represent a range of locations on a picking line starting from, and including, location i and moving in a clockwise direction to, and including, j. Round brackets will exclude the starting or ending points of this range.

Consider a pair of adjacent maximal cuts c_i and c_j and without loss of generality let $i < j$. Then by definition the cuts $c_{i+1}, c_{i+2}, \ldots, c_{j-1}$ cannot be maximal.

In the case where there exists only 1 maximal cut, that is $i = j$. Insert a dummy location next to location i and force all orders picking form location i in the final solution to also pick from the dummy location. A new feasible solution may then be generated such that a maximal cut is not adjacent to itself without increasing the maximal cut.

Consider all the spans passing at least one of the locations in $[i, j]$. All of these spans may be divided into exactly one of 4 sets, (a) the set $\mathcal{A} = \{\langle s_a, e_a\rangle \mid s_a \in (j, i] \text{ and } e_a \in [j, i)\}$, representing those spans which pass both locations i and j; (b) the set $\mathcal{B} = \{\langle s_b, e_b\rangle \mid s_b \in (j, i] \text{ and } i \leq e_b \in [i, j)\}$, representing those spans which pass i and end before j; (c) the set $\mathcal{C} = \{\langle s_c, e_c\rangle \mid s_c \in (i, j] \text{ and } e_c[j, i)\}$, representing those spans that pass j and have a starting position after i; (d) the set $\mathcal{D} = \{\langle s_d, e_d\rangle \mid s_d \in (i, j] \text{ and } e_d \in [i, j)\}$, representing those spans that have both a starting and ending positions between i and j. To illustrate the definitions of these sets, a representative selection of different spans that may fall into each set are shown in Figure 4.8.

For the sets $\mathcal{A}$ and $\mathcal{D}$ there is an equal number of starting and ending positions between the maximal cuts. Therefore it must be shown that there is a equal number of starting positions for the spans in sets $\mathcal{B}$ and $\mathcal{C}$.

Both cuts c_i and c_j are maximal and therefore there are $\bar{C}$ spans passing i and $\bar{C}$ spans passing j. There are $\bar{C} = |\mathcal{A}| + |\mathcal{B}|$ spans which pass location i Similarly there are $\bar{C} = |\mathcal{A}| + |\mathcal{C}|$ spans which pass location j. Therefore $|\mathcal{B}| = |\mathcal{C}|$, which proves the theorem.

□

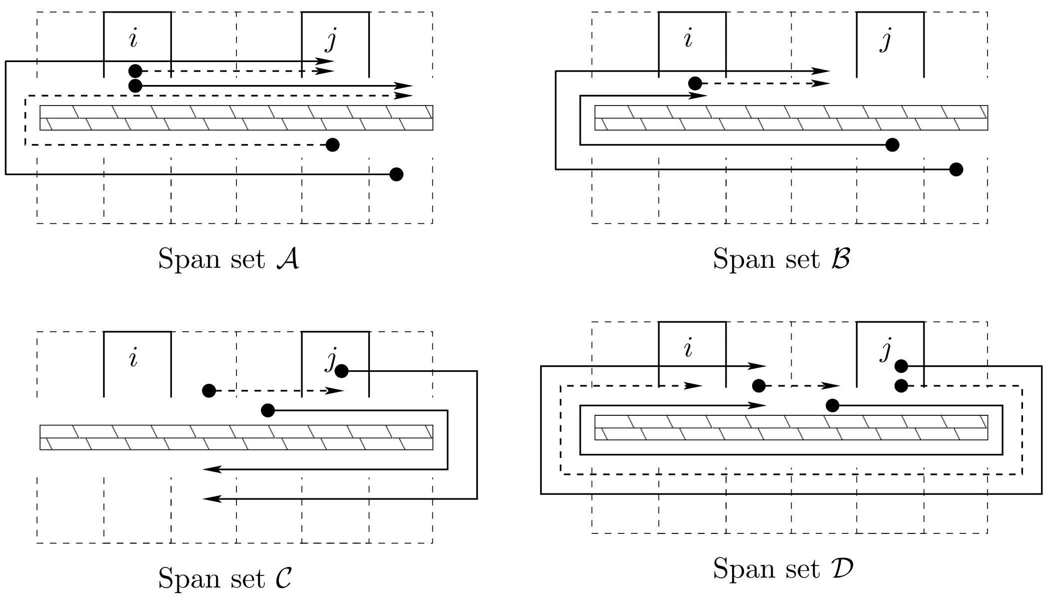


Figure 4.8: *A schematic representation of possible spans for each of the sets $\mathcal{A}$, $\mathcal{B}$, $\mathcal{C}$ and $\mathcal{D}$, where locations i and j are adjacent maximal cuts.*

In Theorem 4 it is shown that the starting and ending positions between two adjacent maximal cuts may be connected without increasing the value of a maximal cut.

Theorem 4. *A solution to formulation* (4.29)–(4.32) *may be transformed to a feasible solution to formulation* (4.20) *to* (4.24) *without increasing the objective function value resulting in* $\bar{C} = C$.

Proof. Consider a pair of adjacent maximal cuts c_i and c_j and without loss of generality let all the cuts c_w with $w \in (i \circlearrowright j)$ be non-maximal. The case where there exists only 1 maximal cut may be handled as in Theorem 3.

Consider the set of starting positions $\{s_q \mid s_q \in (i \circlearrowright j]\}$ and the set of ending ending positions $\{e_q \mid e_q \in [i \circlearrowright j)\}$ order both sets in increasing order. From

Theorem 3 it follows that $|\hat{\mathcal{S}}| = |\hat{\mathcal{E}}| \geq |\mathcal{B}| = |\mathcal{C}|$. Consider any two corresponding ordered elements $e_r \in \hat{\mathcal{E}}$ and $s_r \in \hat{\mathcal{S}}$, where $e_r = w$ and $w \in [i \circlearrowright j)$. It must be shown that $e_r < s_r$.

Consider the set of orders $\mathcal{G} = \{\langle s_g, e_g \rangle \mid s_g \in (i \circlearrowright w]$ and/or $e_g \in [i \circlearrowright w]\}$. These orders may be divided into exactly one of 4 subsets namely, (a) the set $\mathcal{A}^* = \{\langle s_\alpha, e_\alpha \rangle \mid s_\alpha \in (i \circlearrowright w]$ and $e_\alpha \in [i \circlearrowright w]\}$; (b) the set $\mathcal{B}^* = \{\langle s_\beta, e_\beta \rangle \mid s_\beta \in (j \circlearrowright i]$ and $e_\beta \in [i \circlearrowright w]\}$; (c) the set $\mathcal{C}^* = \{\langle s_\gamma, e_\gamma \rangle \mid s_\gamma \in (i \circlearrowright w]$ and $e_\gamma \in [j \circlearrowright i)\}$; (d) the set $\mathcal{D}^* = \{\langle s_\delta, e_\delta \rangle \mid s_\delta \in (i \circlearrowright w]$ and $e_\delta \in [w \circlearrowright j)\}$. Further define the set $\mathcal{B}' = \mathcal{B} \setminus \mathcal{B}^*$.

Assume that $e_r \geq s_r$, then there are more starting positions $\{s_g \mid s_g \in (i \circlearrowright w]\}$ than ending positions $\{e_g \mid e_g \in (i \circlearrowright w]\}$. Therefore $|\mathcal{A}^*| + |\mathcal{C}^*| + |\mathcal{D}^*| \geq |\mathcal{A}^*| + |\mathcal{B}^*|$. Further more $|\mathcal{B}| = |\mathcal{B}^*| + |\mathcal{B}'|$, implying that $|\mathcal{C}^*| + |\mathcal{D}^*| \geq |\mathcal{B}| - |\mathcal{B}'|$ and $|\mathcal{C}^*| + |\mathcal{D}^*| + |\mathcal{B}'| \geq |\mathcal{B}|$.

There are $|\mathcal{C}^*| + |\mathcal{D}^*| + |\mathcal{B}'|$ spans $\langle s_g, e_g \rangle \in \mathcal{G}$ passing location w. In addition there are $\bar{C} - |\mathcal{B}|$ spans $\langle s_a, e_a \rangle \notin \mathcal{G}$ which pass all locations in $[i \circlearrowright j]$ and thus location w. Therefore, $c_w = |\mathcal{C}^*| + |\mathcal{D}^*| + |\mathcal{B}'| + \bar{C} - |\mathcal{B}| \geq \bar{C}$ implying that c_w is at least as large as a maximal cut which contradicts the assumption that c_i and c_j are adjacent maximal cuts. Therefore $e_r < s_r$ for any two corresponding elements $e_r \in \hat{\mathcal{E}}$ and $s_r \in \hat{\mathcal{S}}$.

Consider the matching where each element in $\hat{\mathcal{E}}$ is matched with its corresponding element in $\hat{\mathcal{S}}$. Let the span of the order associated with each starting position be extended (backwards) to the location following the ending position with which it is paired. This new set of spans now satisfies constraint set (4.22) and constraint (4.23). Consider all the paths which is created by connecting these new spans associated with each matching. Each path must begin with a span which passes location i. If a path starts with another span there would be an unmatched starting position in $\hat{\mathcal{S}}$. Similarly each path should end with a span which passes location j. There are only $|\mathcal{B}|$ such starting and ending positions and therefore $|\mathcal{B}|$ such paths. Each path passes all locations in $[i \circlearrowright j]$ and with the addition of the $\bar{C} - |\mathcal{B}|$ spans which passes all locations in $[i \circlearrowright j]$ each cut will have a value of $\bar{C}$. If this process is done for all adjacent cuts a feasible solution is found for the maximal cut formulation. Therefore an optimal solution to the revised maximal cut formulation may be transformed to a feasible solution for the maximal cut formulation. Furthermore, $\bar{C} = C$ because formulation (4.29)–(4.32) is a relaxation of formulation (4.20) to (4.24).

□

Although the complexity of the maximal cut formulation may be reduced the computational time required is still comparatively the same when solving the problem by means of Lingo 11. This may be due to the large data sets which need to be read into the model. The reduced complexity does, however, simplify the implementation of metaheuristic methods as the problem may now be translated to a special case of an assignment problem.

4.5.3 Metaheuristic methods

A metaheuristic method which may be described as a local search with memory structures (LS) was developed. The algorithm is given an initial solution, which consists of a set of spans, and then attempts to search in the local neighbourhood for solutions which either reduce the value of the maximal cut or reduce the number of maximal cuts. The neighbourhood of a solution may be seen as all solutions where a single order is assigned a new span. Following the results in Theorem 4 the size of the neighbourhood is reduced by only considering spans where the order requires the SKU at the starting location. For example, an order requiring 2 SKUs each with 1 location will only have 2 possible starting positions. If during the local search no neighbour may be found which either reduces the value of the maximal cut or reduces the number of maximal cuts a random neighbour is selected to move out of the local optimum. The general framework of the algorithm is described in Algorithm 4.

Algorithm 4: A local search algorithm for solving the maximal cut formulation

```
Data: An initial solution to the revised maximal cut problem
while Best solution remained unchanged for less than a set number of iterations do
    for Each neighbour of the current solution do
        if The neighbour reduces the value of the maximal cut then
            Update both the current and best neighbour;
            break for;
        end
        else if The neighbour has the greatest reduction in the number of maximal cuts then
            Store this neighbour;
        end
    end
    if No neighbour reduces the value of the maximal cut then
        if No neighbour reduces the number of maximal cuts then
            Update the current solution with a random solution;
        end
        else
            Update the current solution with the neighbour with the greatest reduction in the number of
            maximal cuts;
        end
    end
end
```

In addition a hybrid method (HM) variation of Algorithm 4 was developed where variation G4 of Algorithm 3 is called first and the resulting solution is then translated to be given to Algorithm 4 as an initial solution. These two algorithms were tested using the same scenarios as in §4.4 and compared to a variation G4 of Algorithm 3. Testing was done using an Intel(R) Core(TM)2 Duo 3GHz with 3.7 GB ram running UBUNTU 9.10 [31]. All programming was coded in JAVA [30] A summary of the solutions obtained for the greedy approach, local search and hybrid method is given in Table 4.6.

The results in Table 4.6 are graphically summarised in Figure 4.9. These results suggest that the HM variation of Algorithm 4 is the best algorithm with the HM variation of Algorithm 4 achieving the best solution for all but two data sets and achieving a solution that is on average 2% above the maximal cut. For data sets A and K variation LS of Algorithm 4 has a better solution than the HM variation thereof. This may be attributed to a poor initial solution generated by variation G4 of Algorithm 3. The results suggest that a good initial solution significantly improves the solution quality of variation LS of Algorithm 4.

Data set	Size (n, SKUs, m)	G4	LS	HM	Maximal cut
A	(1262, 49, 49)	1252	1232*	1233	1232
B	(1264, 54, 54)	1241	1241	1226*	1226
C	(1265, 51, 51)	1185	1185	1162*	1161
D	(1263, 56, 56)	1133	1141	1086*	1072
E	(1264, 51, 51)	1111	1119	1069*	1069
F	(1258, 53, 53)	1049	1055	1013*	1005
G	(1260, 56, 56)	1013	1013	967*	955
H	(1244, 54, 54)	1030	1041	999*	992
I	(1264, 56, 56)	977	978	955*	947
J	(1258, 55, 55)	1067	1063	1031*	1025
K	(943, 63, 63)	290	274*	292	259
L	(846, 56, 56)	245	238	234*	232
M	(728, 51, 51)	189	177	152*	152
N	(396, 63, 63)	140	115	93*	90
O	(733, 55, 55)	150	130	125*	125
P	(242, 64, 64)	58	50	45*	45
Q	(574, 48, 48)	109	97	81*	80
R	(90, 48, 48)	9	9	7*	7
S	(158, 55, 55)	22	17	14*	14
T	(82, 51, 51)	12	11	8*	8
U	(80, 56, 56)	8	8	6*	6
V	(89, 42, 42)	11	10	9*	9

Table 4.6: *A comparison of the number of cycles traversed between the G4 variation of Algorithm 3 and the local search (LS) and hybrid method (HM) variations of Algorithm 4 for a set of OSP instances. The lower bound for each instance is given as the maximal cut with the shortest solution obtained by a heuristic or metaheuristic algorithm indicated by an asterisk. An upper bound for each instance may be seen as the number of orders n.*

The results for the OSP instances with duplicated SKUs presented in Table 4.7 are consistent with those where no duplicated SKUs are present. The HM variation of Algorithm 4 shows the best performance for all data sets and on average achieves a solution within 2% of the maximal cut.

Although the HM variation of Algorithm 4 is shown to outperform variation G4 of Algorithm 3 in terms of solution quality the computational times for each algorithm must also be taken into account when considering an approach for use when solving the SLP. Solution procedures for solving the SLP will need to use an OSP algorithm to compute the quality of a SKU configuration on a picking line. This may have to be performed multiple times when solving the SLP. Table 4.8 summarises the computational times for the different algorithms.

The results in Tables 4.8 and 4.9 illustrates a significant difference in computational times between variation G4 of Algorithm 3 and variations LS and HM of Algorithm 4. Variation G4 of Algorithm 3 consistently finishes within 0.7 of a second where variations LS and HM of Algorithm 4 requires on average 6 and 17 seconds respectively. It should be noted that passing a good initial solution to variation HM strengthens the solution but results in a computational time which is significantly larger than the standard LS variation. This increase in computational time is too large to be explained by the addition of variation G4 of Algorithm 3 alone and must be attributed by reaching a weak local optimum quickly. This hypothesis is strengthened by the results of the random approach suggesting that the good solutions lie within a small region of solution space. All the methods do, however, show significantly shorter computational times than the exact solution to the maximal cut approach, suggesting the use of these algorithms in

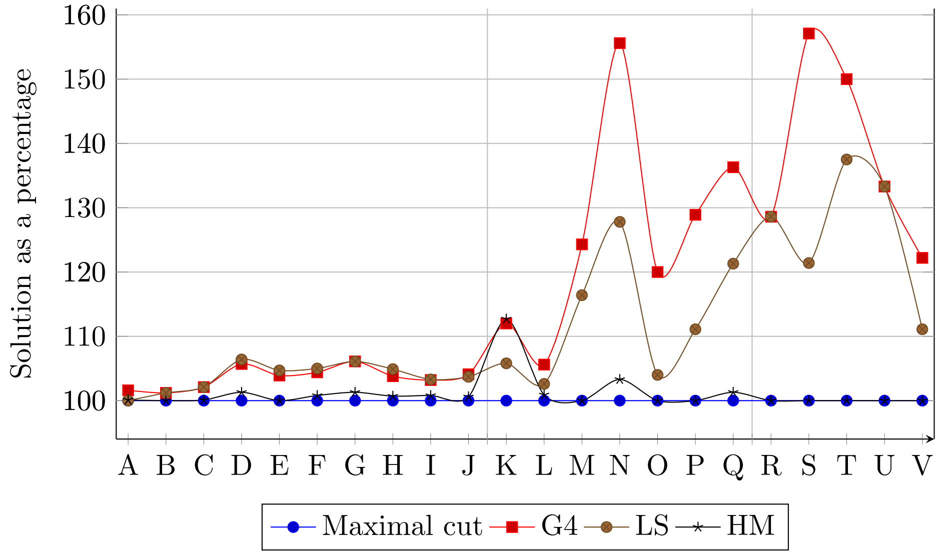


Figure 4.9: *A graphical illustration of the comparison between the G4 variation of Algorithm 3, the local search (LS) and hybrid method (HM) variations of Algorithm 4 and the maximal cut approach for historical data sets. The maximal cut approach represents the 100% benchmark.*

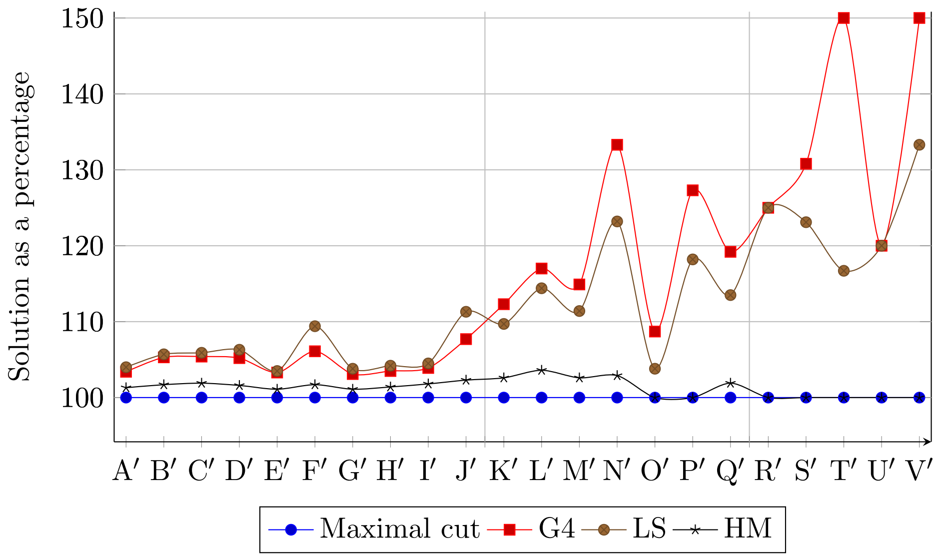


Figure 4.10: *A graphical illustration of the comparison between the G4 variation of Algorithm 3, the local search (LS) and hybrid method (HM) variations of Algorithm 4 and the maximal cut approach for OSP instances where duplicated SKUs are present. The maximal cut approach represents the 100% benchmark.*

Data set	Size (n, SKUs, m)	G4	LS	HM	Maximal cut
A′	(1262, 49, 59)	956	962	937*	925
B′	(1264, 54, 64)	980	984	947*	931
C′	(1265, 51, 61)	925	930	895*	878
D′	(1263, 56, 66)	958	968	926*	911
E′	(1264, 51, 61)	919	921	900*	890
F′	(1258, 53, 63)	950	979	910*	895
G′	(1260, 56, 66)	928	934	910*	900
H′	(1244, 54, 64)	978	985	958*	945
I′	(1264, 56, 66)	933	938	914*	898
J′	(1258, 55, 65)	838	866	796*	778
K′	(943, 63, 73)	302	295	276*	269
L′	(846, 56, 66)	227	222	201*	194
M′	(728, 51, 61)	131	127	117*	114
N′	(396, 63, 73)	92	85	71*	69
O′	(733, 55, 65)	113	108	104*	104
P′	(242, 64, 74)	70	65	55*	55
Q′	(574, 48, 58)	62	59	53*	52
R′	(90, 48, 58)	5	5	4*	4
S′	(158, 55, 65)	17	16	13*	13
T′	(82, 51, 61)	9	7	6*	6
U′	(80, 56, 66)	6	6	5*	5
V′	(89, 42, 52)	9	8	6*	6

Table 4.7: *A comparison of the number of cycles traversed between the G4 variation of Algorithm 3 and the local search (LS) and hybrid method (HM) variations of Algorithm 4 for a set of OSP instances. The lower bound for each instance is given as the maximal cut with the shortest solution obtained by a heuristic or metaheuristic algorithm indicated by an asterisk. An upper bound for each instance may be seen as the number of orders n.*

solving the SLP if the algorithm is called often.

4.6 A relaxation of the OSP

Assumption 6 made in §4.1 states that a picker may not physically pick the first SKU of a new order from the same location as the last pick of the previous order. This assumption was made because management considers it a risk for pickers to pick sequentially from the same location as pick inaccuracies may occur. Because this assumption is based on a managerial decision a relaxed OSP (OSPRX) will be investigated where this assumption is removed.

To use the algorithms discussed in the preceding chapters only a minor change is needed to be done when calculating the spans of orders and their lengths. Let us represent the span of an order in the OSPRX as $\hat{S}_k^i$. Furthermore $|\hat{S}_k^i|$ is now defined by the formula

$$|\hat{S}_k^i| = |\langle i, \hat{e}_k^i \rangle| = \begin{cases} 0 & \text{if } i = \hat{e}_k^i \text{ and} \\ (\hat{e}_k^i - i + m) \mod m & \text{otherwise.} \end{cases} \tag{4.33}$$

These changes may be described with the aid of an example. Consider the the picking line with the following configuration P_1-P_2-P_3-P_1-P_4-P_5 and the order k requiring the SKUs P_1, P_2, P_4 and order l requiring SKU P_5. A schematic representation of this layout is given in Figure 4.11. The spans for the two orders given arbitrary starting positions are given in Table 4.10 for both

Data set	Size (n, SKUs, m)	G4 (ms)	LS (ms)	HM (ms)	Maximal cut (ms)
A	(1262, 49, 49)	534	5853	6376	133310
B	(1264, 54, 54)	487	5825	9931	243830
C	(1258, 55, 55)	384	3825	15259	253480
D	(1265, 51, 51)	408	7061	5812	193450
E	(943, 63, 63)	143	517	1099	202770
F	(1260, 56, 56)	447	4073	13412	303780
G	(1263, 56, 56)	462	5257	17282	464020
H	(1244, 54, 54)	479	4565	24188	253660
I	(1264, 56, 56)	429	3576	10628	563610
J	(396, 63, 63)	279	615	715	31440
K	(1264, 51, 51)	644	3950	22290	14733610
L	(1258, 53, 53)	446	4761	25513	223530
M	(728, 51, 51)	74	635	751	378049
N	(90, 48, 48)	4	19	17	61061
O	(242, 64, 64)	37	161	168	60984
P	(158, 55, 55)	10	48	54	60413
Q	(82, 51, 51)	10	29	24	60272
R	(846, 56, 56)	112	437	541	181486
S	(80, 56, 56)	18	27	33	60113
T	(574, 48, 48)	34	90	176	120533
U	(733, 55, 55)	69	173	325	181071
V	(89, 42, 42)	4	18	12	6096

Table 4.8: *The computational times in milliseconds of the maximal cut, the G4 variation of Algorithm 3 and the local search (LS) and hybrid method (HM) variations of Algorithm 4 for several historical OSP instances where no duplicated SKUs are present.*

Data set	Size (n, SKUs, m)	G4 (ms)	LS (ms)	HM (ms)	Maximal cut (ms)
A′	(1262, 49, 59)	604	16445	27549	152817
B′	(1264, 54, 64)	582	13306	18281	141110
C′	(1258, 55, 65)	448	11089	35295	141770
D′	(1265, 51, 61)	495	13476	19173	961380
E′	(943, 63, 73)	170	1046	1275	161853
F′	(1260, 56, 66)	545	10702	22727	723369
G′	(1263, 56, 66)	559	14718	31841	423106
H′	(1244, 54, 64)	582	7874	32484	362844
I′	(1264, 56, 66)	523	7026	14335	602825
J′	(396, 63, 73)	125	657	612	121003
K′	(1264, 51, 61)	620	9209	23408	483140
L′	(1258, 53, 63)	527	11896	41311	242859
M′	(728, 51, 61)	78	776	771	123169
N′	(90, 48, 58)	9	29	32	61060
O′	(242, 64, 74)	42	406	484	60895
P′	(158, 55, 65)	13	49	62	60445
Q′	(82, 51, 61)	6	29	37	60304
R′	(846, 56, 66)	136	1383	1432	241892
S′	(80, 56, 66)	17	36	43	60159
T′	(574, 48, 58)	41	379	334	60752
U′	(733, 55, 65)	78	392	384	241605
V′	(89, 42, 52)	5	26	25	60127

Table 4.9: *The computational times in milliseconds of the maximal cut, the G4 variation of Algorithm 3 and the local search (LS) and hybrid method (HM) variations of Algorithm 4 for several different OSP instances where duplicated SKUs are present.*

the OSP and OSPRX. It should be noted that for the OSPRX the maximum possible span length is $m - 1$ with a minimum span length of 0.

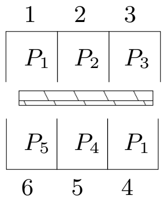


Figure 4.11: *A schematic representation of the layout of an example instance with 6 locations and 5 SKUs.*

OSP			OSPRX		
S_k^2	$=$	$\langle 2, 2\rangle$	$\lvert\hat{S}_k^2\rvert$	$=$	$\langle 2, 4\rangle$
$\lvert S_k^2\rvert$	$=$	6	$\lvert\hat{S}_k^2\rvert$	$=$	2
S_l^6	$=$	$\langle 6, 6\rangle$	$\lvert\hat{S}_l^6\rvert$	$=$	$\langle 6, 6\rangle$
$\lvert S_l^6\rvert$	$=$	6	$\lvert\hat{S}_l^6\rvert$	$=$	0

Table 4.10: *A comparison of the calculation of spans and lengths thereof between the order sequencing problem (OSP) and the relaxed order sequencing problem (OSPRX).*

Furthermore the maximal cut formulation may be used to solve the OSPRX by using the concept of a relaxed cut.

Definition 5. *The relaxed cut of a location i is the number of spans passing location i, excluding the spans starting at location i.*

To solve the OSPRX using the maximal cut formulation, let

$$x_{ik} = \begin{cases} 1 & \text{if order } k \text{ starts at location } i \\ 0 & \text{otherwise,} \end{cases}$$

and

$\hat{C}$ be the relaxed maximal cut.

The following parameters are set in the model. Let

n be the total number of orders,

m be the total number of locations,

$$\hat{d}_{ikj} = \begin{cases} 1 & \text{if order } k \text{ starting at location } i \text{ passes location } j \\ 0 & \text{otherwise} \end{cases}$$

and

$$e_{ikj} = \begin{cases} 1 & \text{if order } k \text{ starting at location } i \text{ is completed at location } j \\ 0 & \text{otherwise.} \end{cases}$$

In terms of these symbols the objective is to

$$\text{minimise} \quad C \tag{4.34}$$
subject to

$$\sum_{i=1}^{m} x_{ik} = 1 \qquad k = 1, \ldots, n, \tag{4.35}$$

$$\sum_{i=1}^{m}\sum_{k=1}^{n} \hat{d}_{ikj} x_{ik} \leq C \qquad j = 1, \ldots, m, \tag{4.36}$$

$$\sum_{k=1}^{n} x_{j+1,k} - \sum_{i=1}^{m}\sum_{k=1}^{n} x_{ik} e_{ikj} = 0 \qquad j = 1, \ldots, m-1, \tag{4.37}$$

$$\sum_{k=1}^{n} x_{1k} - \sum_{i=1}^{m}\sum_{k=1}^{n} x_{ik} e_{ikm} = 0, \tag{4.38}$$

$$x_{ik} \in \{0, 1\} \qquad i = 1, \ldots, m, \quad k = 1, \ldots, n. \tag{4.39}$$

The parameter $\hat{d}_{ikj}$ is generated by assigning a value of 1 to all the locations in the span of order k starting at location i excluding location i. The same example as above is used with a picking line with the following configuration P_1-P_2-P_3-P_1-P_4-P_5 and orders k requiring the SKUs P_1, P_2, P_4 and l requiring SKU P_5. A schematic representation of this layout is given in Figure 4.11. If order k starts at location 2 then $\hat{d}_{2k3} = \hat{d}_{2k4} = \hat{d}_{2k5} = 1$ and $\hat{d}_{2k1} = \hat{d}_{2k2} = \hat{d}_{1k6} = 0$ must be generated. Similarly if order l starts at location 5 then $\hat{d}_{5l6} = 1$ and $\hat{d}_{5l1} = \hat{d}_{5l2} = \hat{d}_{5l3} = \hat{d}_{5l4} = \hat{d}_{5l5} = 0$ must be generated.

To determine the difference in number of cycles traversed between the OSP and OSPRX both problems were solved with their respective maximal cut formulations. Tables 4.11 and 4.12 summarise the results.

The results in Table 4.11 suggest a significant difference in the number of cycles traversed when solving the OSPRX compared to the OSP where duplicated SKUs are not present. The number of cycles traversed for the OSPRX are in total 10.4% less than the number of cycles traversed for the OSP. The results in Table 4.12 also display a significant difference in number of cycles traversed between the OSPRX and the OSP where duplicated SKUs are present. The solutions to the OSPRX are in total 3% less than the solutions to the OSP where duplicated SKUs are present which is significantly smaller than the cases where duplicated SKUs are present. These results suggest that significant improvements in cycles traversed may be achieved if assumption 6 is removed from the definition of the OSP and suggests a reconsideration by Pep as to its inclusion.

4.7 Chapter Summary

The OSP was modelled as an equality generalized travelling salesman problem and an exact formulation was presented. The computational effort required to solve the exact formulation is too large for real life data sets. Not even a linear relaxation of the model could be solved. This led to the investigation of heuristic and metaheuristic methods.

To determine the effectiveness of these algorithms a good lower bound was necessary. This was achieved by reducing the problem size and making use of the concept of a maximal cut. This

Data set	Size (n, SKUs, m)	OSP	OSPRX	Percentage improvement
A	(1262, 49, 49)	1232	1113	10.69%
B	(1264, 54, 54)	1226	966	26.92%
C	(1265, 51, 51)	1161	998	16.33%
D	(1263, 56, 56)	1072	979	9.5%
E	(1264, 51, 51)	1069	970	10.21%
F	(1258, 53, 53)	1005	945	6.35%
G	(1260, 56, 56)	955	918	4.03%
H	(1244, 54, 54)	992	953	4.09%
I	(1264, 56, 56)	947	909	4.18%
J	(1258, 55, 55)	1025	925	10.81%
K	(943, 63, 63)	259	245	5.71%
L	(846, 56, 56)	232	199	16.58%
M	(728, 51, 51)	152	133	14.29%
N	(396, 63, 63)	90	85	5.88%
O	(733, 55, 55)	125	110	13.64%
P	(242, 64, 64)	45	42	7.14%
Q	(574, 48, 48)	80	70	14.29%
R	(90, 48, 48)	7	7	0%
S	(158, 55, 55)	14	14	0%
T	(82, 51, 51)	8	8	0%
U	(80, 56, 56)	6	6	0%
V	(89, 42, 42)	9	8	12.5%

Table 4.11: *A comparison of the number of cycles traversed between the order sequencing problem (OSP) and the relaxed order sequencing problem (OSPRX) for different instances where duplicated SKUs are not present. In both cases the maximal cut approach was used.*

Data set	Size (n, SKUs, m)	OSP	OSPRX	Percentage improvement
A′	(1262, 49, 59)	925	900	2.78%
B′	(1264, 54, 64)	931	909	2.42%
C′	(1265, 51, 61)	878	857	2.45%
D′	(1263, 56, 66)	911	880	3.52%
E′	(1264, 51, 61)	890	873	1.95%
F′	(1258, 53, 63)	895	858	4.31%
G′	(1260, 56, 66)	900	881	2.16%
H′	(1244, 54, 64)	945	915	3.28%
I′	(1264, 56, 66)	898	879	2.16%
J′	(1258, 55, 65)	778	744	4.57%
K′	(943, 63, 73)	269	256	5.08%
L′	(846, 56, 66)	194	188	3.19%
M′	(728, 51, 61)	114	109	4.59%
N′	(396, 63, 73)	69	66	4.55%
O′	(733, 55, 65)	104	101	2.97%
P′	(242, 64, 74)	55	53	3.77%
Q′	(574, 48, 58)	52	49	6.12%
R′	(90, 48, 58)	4	4	0%
S′	(158, 55, 65)	13	13	0%
T′	(82, 51, 61)	6	6	0%
U′	(80, 56, 66)	5	5	0%
V′	(89, 42, 52)	6	5	20%

Table 4.12: *A comparison of the number of cycles traversed between the order sequencing problem (OSP) and the relaxed order sequencing problem (OSPRX) for different instances where duplicated SKUs are present. In both cases the maximal cut approach was used.*

lower bound or maximal cut approach was shown to be within 1 cycle of an optimal solution to the OSP and an algorithm was developed to transform the solution of the maximal cut approach to a feasible solution to the OSP by increasing the number of cycles traversed by at most 1 cycle. When introducing multiple pickers the maximal cut was shown to be applicable in practice. The computational time required to solve the problem exactly, however, did not allow for the use in solving the SLP and thus other heuristics were developed.

Four variations of a greedy algorithm were developed, each with a distance measure of the next best order. These four variations were compared and it was found that a relative distance measure that takes the minimum span and the number of picks in an order into account was the best measure (G4). A random local search which required an initial solution was also developed to solve the maximal cut formulation. A random solution (LS) and a solution generated by means of the greedy approach (HM) were tested. It was found that the HM variation of this local search approach which made use of an initial solution from the greedy variation G4 showed the best performance. The solutions thereof were on average 2% off of the maximal cut approach.

Computation time was also taken into account when testing the heuristic and metaheuristic approaches and all approaches had significantly better computational times than the maximal cut approach. The greedy algorithms took at most 1 second while the LS and HM methods took slightly longer on average 6-17 seconds depending on whether duplicated SKUs were present. It was concluded that both the G4 variation of the greedy algorithm and the HM variation of the local search approach could be used when solving the SLP depending on how often the algorithm would be called in SLP solution procedures.

In the definition of the OSP an assumption was included based on a managerial decision. This assumption ensured that a picker would not be able to pick at the same location consecutively without traversing an entire cycle. The effects of this assumption were investigated by defining a relaxed OSP(OSPRX). It was found that the number of cycles traversed when solving the OSPRX was significantly smaller than when solving the OSP. It was also found that the differences in number of cycles traversed between the two problems is significantly smaller for cases where duplicated SKUs are present.

CHAPTER 5

SKU location problem

Contents

The order pick operation consist of three tiers of decisions. Three subproblems, namely the Picking line allocation problem (PLAP), SKU location problem (SLP) and the Order sequencing problem (OSP) were identified, each of which focus' on a different decision tier. The OSP was discussed in Chapter 4.

The SKU location problem (SLP) may be described as the assignment of locations to all the SKUs already allocated to a picking line such that after sequencing the orders the picking time/distance is minimised. Pep currently does not allow for a SKU to be allocated to multiple locations in a picking line, but after consulting with management it was decided that the feasibility of duplicating SKUs on a picking line should be investigated. The duplication of SKUs was therefore included in the formulation of the SLP. The determination of which SKUs should be duplicated as well as the number of additional locations which should be allocated to them, is beyond the scope of the SLP.

To model the SLP the following assumptions need to be made:

1. All assumptions used for modelling and solving the OSP in §4.1 still hold.

2. All locations are identical and the time required to pick a SKU is independent of other SKU locations.

3. Any two SKUs may be placed adjacent to each other.

Using the above mentioned assumptions the objective of the SLP may be transformed to the minimisation of the number of cycles traversed to complete a picking line, as for the OSP. Modelling the SLP as well as solution methods thereof will be discussed for the remainder of this chapter.

5.1 A mathematical model for the SLP

Let a picking line consist of a set of SKUs $\mathcal{M}$ and duplicated SKUs $\mathcal{M}^d$, such that $|\mathcal{M}|+|\mathcal{M}^d| = m$, the number of locations in the picking line. Each SKU ($\varsigma_t \in \mathcal{M} \cap \mathcal{M}^d$) must be assigned to a location while minimising the total time required to pick all orders. Let the notation $\varsigma_t \leftrightarrow \varsigma_r$ indicate that if an order requires ς_t it may pick ς_r and *vice versa*. ς_t and ς_r would therefore both represent the same physical product, but each in a different location. Let the ordered set $\mathcal{P}$ represent the solution to the SLP, with the element $\varrho_i \in \mathcal{P}$ representing the SKU in location i. A solution to the SLP will be represented as an assignment of locations only as the actual sequence of orders may be generated separately by solving the corresponding OSP.

The following definition is required to model the problem exactly. Let a pick be the act of picking a SKU in a picking line for an order. A pick may thus be performed at any of the locations which contains the required SKU. The SLP may be viewed as a joint assignment problem and clustered travelling salesman problem (CTSP). A CTSP may be seen as a variant of the TSP where certain nodes must be visited consecutively. The SLP therefore simultaneously assigns locations to SKUs and sequences the picks using a CTSP structure. Thus each SKU is assigned a location and a CTSP is solved for each possible assignment of bay locations. To formulate the SLP let

$$x_{ijkl} = \begin{cases} 1 & \text{if pick } i \text{ positioned at location } j \text{ is followed by pick } k \text{ positioned at location } l \\ 0 & \text{otherwise,} \end{cases}$$

$$\varsigma_{tj} = \begin{cases} 1 & \text{if SKU } t \text{ is positioned at location } j \\ 0 & \text{otherwise,} \end{cases}$$

p_i^p be the position of pick i within the sequence,

p_j^s be the SKU at location j within the picking line.

The following parameters are used in the model. Let

$\mathcal{O}_i$	be the set of picks in order i,
$\lvert\mathcal{O}_i\rvert$	be the number of picks in order i,
$\mathcal{I}_i$	be the set of SKUs corresponding to pick i,
η	be the total number of picks,
m	be the total number of locations, and
$\bar{d}_{jl}$	be the number of locations between location j and l

The objective is then to

$$\text{minimise} \quad \sum_{i=1}^{\eta}\sum_{j=1}^{m}\sum_{k=1}^{\eta}\sum_{l=1}^{m} \bar{d}_{jl} x_{ijkl} \tag{5.1}$$

subject to

$$\sum_{i=1}^{\eta}\sum_{j=1}^{m}\sum_{l=1}^{m} x_{ijkl} = 1 \qquad \forall\ k, \tag{5.2}$$

$$\sum_{j=1}^{m}\sum_{k=1}^{\eta}\sum_{l=1}^{m} x_{ijkl} = 1 \qquad \forall\ i, \tag{5.3}$$

$$p_i^p - p_k^p + \eta \sum_{j=1}^{m}\sum_{l=1}^{m} x_{ijkl} \leq \eta - 1 \qquad \forall\ i, k, k \neq 1, \tag{5.4}$$

$$p_i^p - p_k^p \leq |\mathcal{O}_j| \qquad \text{if pick } i \text{ and } k \text{ are both in branch order } j \tag{5.5}$$

$$x_{ijkl} \leq 0.5 \left(\sum_{t \in \mathcal{I}_i} \varsigma_{tj} + \sum_{r \in \mathcal{I}_k} \varsigma_{rl} \right) \qquad \forall\ i, k \tag{5.6}$$

$$p_j^s = \sum_{t=1}^{m} t \varsigma_{tj} \qquad \forall\ j, \tag{5.7}$$

$$\sum_{t=1}^{m} \varsigma_{tj} = 1 \qquad \forall\ j, \tag{5.8}$$

$$x_{ijkl} \in \{0, 1\} \qquad \forall\ i, j, k, l, \tag{5.9}$$

$$\varsigma_{tj} \in \{0, 1\} \qquad \forall\ t, j, \tag{5.10}$$

$$p_i^p \geq 0 \qquad \forall\ i, \tag{5.11}$$

$$p_j^s \geq 0 \qquad \forall\ j. \tag{5.12}$$

The objective function (5.1) minimises the total distance travelled to complete the list of picks. Equation sets (5.2) and (5.3) ensure that each pick has only two adjacent picks one before and one after. Equation set (5.4) ensures that no subtours occur between picks by adapting the MTZ constraints discussed in Punnen [27]. Equation set (5.5) ensures that all picks from the same order are completed before picks from another order may begin. Equation set (5.6) ensures that a pick may only take place at a location if the SKU associated with that pick is placed at that location. It should be noted that a pick could require either one of the duplicated SKUs if present. Equation set (5.7) assigns a location to each SKU with equation set (5.8) ensuring that each location is assigned a single SKU. For a standard size instance faced by Pep the number of variables are in excess of 550×10^9 and the number of constraints in excess of 180×10^6.

Due to the size of the data sets from Pep as well as the complexity of the formulation exact solution methods for the problem were not be investigated. Heuristic and metaheuristic approaches were investigated further and are discussed in the remainder of this chapter.

To explain the solution approaches that follow in the next sections the following notation is required. Let

$\mathcal{R}(\varsigma_t)$	be the set of orders requiring SKU ς_t and
$\alpha(\varsigma_t, \varsigma_r) = \lvert\mathcal{R}(\varsigma_t) \cap \mathcal{R}(\varsigma_r)\rvert$	be the number of orders requiring both SKU ς_t and ς_r.

Note that $\alpha(\varsigma_t, \varsigma_r) = 0$ if $\varsigma_t \leftrightarrow \varsigma_r$ for use in cases where a SKU is allocated multiple locations as an order should pick from either one of the locations where the SKU is stored.

5.2 Heuristic approaches

Due to the unidirectional carousel structure of the picking line discussed in §4.2 the SLP may be viewed as the assigning of locations to SKUs in a variation of the unidirectional carousel. There is a major difference, however, between this system and carousel systems discussed in literature which requires consideration. Carousel systems in literature do not display a long term finite set of orders. In the picking line all the orders are known for every new assignment of locations to SKUs, while orders are stochastic in the cases of carousel systems considered in literature [16]. Estimated distributions that describe the proportion of orders expected to require a specific SKU are used to optimise these carousel systems. These estimations are usually based on historical data. Although there is a difference in the information available for the two systems, SKU location algorithms used in literature were applied to the problem considered here.

A known heuristic for SKU allocations in bidirectional carousel problems is the so called organ pipe heuristic (OP) [16]. The OP heuristic initially places the most dense SKU in an arbitrary location and then sequentially places the next most dense SKU on alternating sides of the already allocated SKUs. In literature the distributions, and not deterministic data, are used as a measure of a SKUs density. The pseudo code for this heuristic is presented in Algorithm 5. The OP heuristic was shown to be optimal, in the long run, for carousel systems with stochastic orders [16].

Algorithm 5: Organ pipe heuristic

Input: A set $\mathcal{M}$ of SKUs.
Output: A set $\mathcal{P}$ of SKUs

Create an ordered list L from $\mathcal{M}$ according to number of picks where L_i represents the ith element in the list ;
`/* This ordered list is generated such that duplicated SKUs are adjacent */`
for *All* L_i **do**
  if $i \mod 2 = 0$ **then**
    $\varrho_{\lceil \frac{m}{2} \rceil - \frac{i}{2}} \leftarrow L_i$
  end
  else
    $\varrho_{\lceil \frac{m}{2} \rceil + \lfloor \frac{i}{2} \rfloor} \leftarrow L_i$
  end
end

A greedy heuristic (GP) was shown to be optimal for unidirectional carousel systems with stochastic orders and is discussed in Hassini [16]. The algorithm is described in Algorithm 6, where SKUs are sequenced according to density.

Both the OP and GP heuristics initially do not take into account duplicated SKUs, although the OP heuristic can handle duplicates if a SKU is only duplicated once and the sorting algorithm ensures that the duplicated instances of a SKU are adjacent in the sorted list. The two heuristics

Algorithm 6: Greedy allocation heuristic

Input: A set $\mathcal{M}$ of SKUs.
Output: A set $\mathcal{P}$ of SKUs
1 Create an ordered list L from set $\mathcal{M}$ according to the number of picks where L_i represents the ith element in the list ;
2 **for** *All* L_i **do**
3 | $\varrho_i \leftarrow L_i$
4 **end**

were further adapted to handle duplicated SKUs. The OP heuristic was adapted (OPA) by separately generating an organ pipe configuration for the sets $\mathcal{M}$ and $\mathcal{M}^d$ and then joining the two sets. The adaptation of the GP heuristic (GPA) is done in a similar way. These two adaptations may also be used when SKUs are duplicated more than once by generating separate sections for each set of similar SKUs. A schematic representation of an example layout for these different heuristics is given in Figure 5.1. All SKU allocations generated by one of the heuristic approaches were solved using the HM algorithm discussed in §4.5 to determine an order sequence.

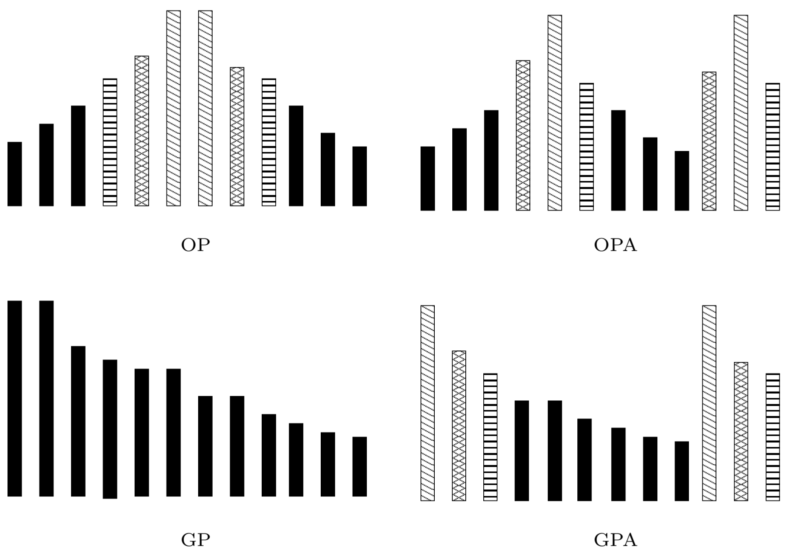


Figure 5.1: *A schematic representation of the locations of SKUs in a picking line for the organ pipe (OP), adapted organ pipe (OPA), greedy (GP) and the adapted greedy (GPA) heuristics. Each bar represents a SKU and the height the density. Bars with the same pattern represent duplicated SKUs. The GP algorithm cannot handle duplicated SKUs.*

5.3 An ant colony approach

Ant colony algorithms are a type of metaheuristic based on the observed behaviour of ants. The first algorithm of this class, the "Ant system" (AS), was designed specifically for the travelling salesman problem.

Ants naturally search for food by sending foragers to randomly look for food. As ants walk they lay down pheromones which is a chemical compound which can be detected by other ants. These pheromones may be seen as markers allowing the ant to find the path back to the colony.

If an ant finds food it follows its own pheromone trail back to the colony and in the process leaves a pheromone trail which is used by other ants to find the same food source. If an ant finds such a pheromone trail and follows it to the food source and back it too lays down pheromones increasing the intensity of the trail until eventually there is a single line of ants walking to and from the food source.

If multiple pheromone trails exist to the same food source it is expected that the shorter trails will receive more additional pheromones by ants that walk along the trail. Therefore it would be expected that the shorter paths to food will grow in pheromone intensity quickly, attracting more ants. In addition trails that do not have frequent passes will loose intensity as the pheromones evaporate. It is the idea that shorter trails will grow in pheromone intensity quicker on which the ant colony algorithms are based. Ant colony algorithms should be constructed where a trail presenting a high intensity of pheromones is a more attractive option than following the shortest local edge.

Ant colonies may be described as a system of self-organisation. A more formal definition is provided by Camazine *et al.* [6]: "Self-organisation is a process in which pattern at the global level of a system emerges solely from numerous interactions among lower-level components of the system. Moreover, the rules specifying interactions among the system's components are executed using only local information, without reference to the global pattern". Ants are not controlled by some central entity but rather each ant follows a simple set of rules, one of which is to follow pheromone trails. The behaviour of the entire system is determined by the small changes to the environment made by each ant.

The SLP may be viewed as an assignment problem where two different assignments are compared by solving the corresponding OSP instance. The AS algorithm was adapted form the Ant System Adaptation (ASA) to make it applicable for this problem. In the ASA algorithm the pheromones, used in the AS, may be translated to the desirability of two SKUs to be placed adjacent to each other. The visibility (ν_{ij}) between two nodes i and j in the AS algorithm depends on the distance (d_{ij}) between the two nodes in a TSP, where $\nu_{ij} = \frac{1}{d_{ij}}$. A concept of distance, in the ASA algorithm, between two SKUs ς_i and ς_j for the SLP is taken as $\frac{n}{\alpha(\varsigma_i,\varsigma_j)}$. This definition implies that a distance (in the TSP sense) corresponds to the the proportion of orders requiring both SKUs.

During each step in the ASA algorithm a SKU is assigned to a location by making use of the random proportional transition rule which may be stated as

$$p_{ij}^k(t) = \begin{cases} \dfrac{(\tau_{ij}(t))^{\alpha} \cdot (\nu_{ij})^{\beta}}{\sum_{\varsigma_l \in \mathcal{U}_i^k} (\tau_{il}(t))^{\alpha} \cdot (\nu_{ij})^{\beta}} & \text{if } \varsigma_j \in \mathcal{U}_i^k \\ 0 & \text{otherwise,} \end{cases} \tag{5.13}$$

where $p_{ij}^k(t)$ is the probability that ant k places SKU j adjacent to SKU i at iteration t, α and β are two parameters controlling the relative importance of the desirability factors, namely the pheromone levels $\tau_{ij}(t)$ and the visibility $\nu_{ij} = \frac{\alpha(\varsigma_i,\varsigma_j)}{n}$ [7]. The set $\mathcal{U}_i^k$ contains unassigned SKUs at the current assignment step. Note that when $\alpha = 0$ only visibility is taken into consideration and when $\beta = 0$ only pheromones are considered influential. The choice of these two parameters needs to generate a good balance between intensity and diversity.

After each iteration, an ant leaves a quantity $\Delta\tau_{ij}^k(t)$ of pheromone between two adjacent SKUs.

This quantity of pheromone is calculated by means of the formula

$$\Delta\tau_{ij}^{k}(t) = \begin{cases} \dfrac{Q}{C^{k}(t)} & \text{if } (i,j) \in \mathcal{P}^{k}(t) \\ 0 & \text{if } (i,j) \notin \mathcal{P}^{k}(t), \end{cases} \tag{5.14}$$

where $\mathcal{P}^k(t)$ is the SLP solution constructed by ant k during iteration t, $C^k(t)$ is the cut generated by solving the associated OSP instance with the greedy heuristic (G3) presented in Algorithm 3, and Q is a parameter of scaling serving as a relative measure of performance [7].

A form of "evaporation" for the sub-optimal solutions occurs after each iteration. The pheromones are updated by means of

$$\tau_{ij}(t+1) = (1-\rho)\cdot \tau_{ij}(t) + \Delta\tau_{ij}, \tag{5.15}$$

where $\Delta\tau_{ij} = \sum_{k=1}^{m} \Delta\tau_{ij}^{k}(t)$ and ρ is a parameter for evaporation [7]. The pseudo code for the complete ASA algorithm is described in Algorithm 7 with the final solution undergoing further optimisation with the HM algorithm discussed in §4.5. All the required parameters were determined by means of simulation and statistical analysis. The results for the parameter testing may be found in Appendix C.1.

Algorithm 7: Ant colony algorithm

Input: A set $\mathcal{M}$ of SKUs and a set $\mathcal{M}^d$ of duplicated SKUs.
Output: A set $\mathcal{P}$ of SKUs

- $C_{\text{best}} = n$;
- **for** $t <$ *iterations* **do**
    - **for** *Each ant* k **do**
        - Generate $\mathcal{P}^k(t)$ from the SKUs in sets $\mathcal{M}$ and $\mathcal{M}^d$ and solve with greedy heuristic;
        - **if** $C^k(t) < C_{\text{best}}$ **then**
            - $C_{\text{best}} \leftarrow C^k(t)$;
            - $\mathcal{P} \leftarrow \mathcal{P}^k(t)$;
        - **end**
    - **end**
    - Update pheromone levels;
- **end**

5.4 Clustering algorithms

Cluster analysis is the art of finding groups in data [20]. Consider a set of objects differing according to several characteristics. The goal of clustering is to from groups of items such that the objects in the same group are similar and those in different groups are dissimilar according to a set of characteristics. Clustering occurs in everyday life and is part of the learning process. For example, children learn to differentiate between cats and dogs even though each cat or dog is different. This process of clustering objects has been applied to data where elements are classified according to a set of properties and arranged into good clusters using different algorithms. These processes only became usable after the development of computers. This clustering of data has been used in many domains such as artificial intelligence, marketing, medical research and economics among others [20].

A main consideration with the SLP problem is the size of the solution space in conjunction with the computational time in solving the resulting OSP. A reduction in size may allow for the viable use of local search techniques. Although clustering algorithms are usually applied in data analysis the basic idea of grouping objects was applied to the SLP. SKUs must be grouped based on some set of SKU properties. Clustering algorithms were thus developed which cluster different SKUs into a set $\mathcal{Q}$ of clusters. The SKUs within each cluster are then forced to remain in a single fixed sequence of adjacent SKUs for each SLP solution.

An agglomerative hierarchical clustering approach was used where each SKU is initially in its own cluster q_i. The two closests clusters are then merged $(q_i \oplus q_j)$ until the number of clusters reaches a threshold $(m^c \leq m)$ [20]. Let us define $d(q_i, q_j)$ as the distance between two clusters q_i and q_j. Algorithm 8 provides the pseudo code for the agglomerative hierarchical clustering procedure.

Algorithm 8: Agglomerative hierarchical clustering

Input: A set $\mathcal{Q}$ of clustered SKUs where $|\mathcal{Q}| = m$
Output: A set $\mathcal{Q}$ of clustered SKUs where $|\mathcal{Q}| = m^c$

```
while |Q| > m^c do
    (q_s, q_t) → arg max_{i,j}(d(q_i, q_j) ∀ q_i, q_j ∈ Q);
    q_s ⊕ q_t;
end
```

Several variations of the operator $d(q_i, q_j)$ were tested. To understand these variations the following notation is required. Let

$\mathcal{R}(q_i)$	be the set of orders requiring a SKU in cluster q_i and
$\alpha(q_i, q_j)$	be the number of orders requiring all SKUs in both clusters q_i and q_j.

To illustrate the calculation of the above notation consider the example clusters given in Table 5.1.

q_1		q_2	
SKU	Orders	SKU	Orders
ς_1	A, B, C	ς_3	A, B
ς_2	B, D, E	ς_4	B, F

Table 5.1: *An example of 2 clusters (q_1 and q_2) each with 2 SKUs and the orders (A, B, C, D, E, F) which require each SKU.*

It follows that $\mathcal{R}(q_1) = \{\text{A, B, C, D, E}\}$ and $\mathcal{R}(q_2) = \{\text{A, B, F}\}$ and $\alpha(q_i, q_j) = 1$ as order B is the only order which requires all the SKUs in each cluster.

The first variation for the distance operator will be referred to as the Maximum Adjacencies variation (MA). The MA variation tries to maximise the number of orders which require all the SKUs in a cluster by using the distance measure

$$d(q_i, q_j) = \begin{cases} \alpha(q_i, q_j) & \text{if } q_i \cap q_j = \emptyset \\ 0 & \text{otherwise} \end{cases} \tag{5.16}$$

Let cluster q_i be dominated by cluster q_j if $\mathcal{R}(q_i) \subseteq \mathcal{R}(q_j)$. If a cluster is dominated then it might be advantages to merge those two clusters. The Adjacency Domination variation (AD) calculates a measure of domination as the proportion of a cluster's orders required by another cluster using the distance measure

$$d(q_i, q_j) = \begin{cases} \dfrac{|\mathcal{R}(q_i) \cap \mathcal{R}(q_j)|}{|\mathcal{R}(q_i)|} & \text{if } q_i \cap q_j = \emptyset \\ 0 & \text{otherwise.} \end{cases} \tag{5.17}$$

Consider again the example in Table 5.1 with $\mathcal{R}(q_1) \cap \mathcal{R}(q_2) = \{\text{A, B}\}$. It follows that $d(q_1, q_2) = \frac{2}{5}$ and $d(q_2, q_1) = \frac{2}{3}$ according to equation (5.17).

Instead of considering adjacencies between clusters it may be beneficial to consider adjacencies between individual SKUs in different clusters. The SKU Adjacency Domination variation (SAD) calculates a consolidated measure of adjacencies between individual SKUs in different clusters. Some of the effects of domination, as discussed earlier, are also taken into account by using the distance measure

$$d(q_i, q_j) = \begin{cases} \dfrac{\sum_{\varsigma_t \in q_i} \sum_{\varsigma_r \in q_j} \frac{\alpha(\varsigma_t, \varsigma_r)}{|\mathcal{R}(\varsigma_t)|}}{|\mathcal{R}(q_i)|} & \text{if } q_i \cap q_j = \emptyset \\ 0 & \text{otherwise,} \end{cases} \tag{5.18}$$

Table 5.2 illustrates the calculation of $d(q_1, q_2)$ according to equation (5.18) where q_1 and q_2 are given in Table 5.1.

ς_t \ ς_r	ς_1	ς_2	ς_3	ς_4
ς_1			$\frac{2}{3}$	$\frac{1}{3}$
ς_2			$\frac{1}{3}$	$\frac{1}{3}$
ς_3	$\frac{2}{2}$	$\frac{1}{2}$		
ς_4	$\frac{1}{2}$	$\frac{1}{2}$		

$$\begin{aligned} |\mathcal{R}(q_1)| &= 5 \\ |\mathcal{R}(q_2)| &= 3 \\ d(q_1, q_2) &= \tfrac{5}{15} \\ d(q_2, q_1) &= \tfrac{5}{6} \end{aligned}$$

Table 5.2: *A table of values used to calculate the distance measure according to equation* (5.18) *for the SAD variation between the two clusters given in the example in Table 5.1. The values in the subtable are calculated with the formula* $\alpha(\varsigma_t, \varsigma_r)/|\mathcal{R}(\varsigma_t)|$.

A similar variation to the SAD was considered where domination is not taken into account. This SKU Adjacency variation (SA) uses the distance measure

$$d(q_i, q_j) = \begin{cases} \dfrac{\sum_{\varsigma_t \in q_i} \sum_{\varsigma_r \in q_j} \frac{\alpha(\varsigma_t, \varsigma_r)}{|\mathcal{R}(\varsigma_t)|}}{|\mathcal{R}(q_i)| + |\mathcal{R}(q_j)|} & \text{if } q_i \cap q_j = \emptyset \\ 0 & \text{otherwise.} \end{cases} \tag{5.19}$$

Once clusters of SKUs have been created the solution structure becomes smaller and allows for easier use of heuristic and metaheuristic approaches as each cluster is viewed as a single entity during the assignment of SKUs to locations. Several sequencing approaches were further developed to take advantage of this solution structure.

5.4.1 Random search

The random search of clusters (RSC) randomly allocates a position for each cluster within the picking line. The philosophy of clustering suggests that all the SKUs in a cluster should be placed in close proximity to each other. All the SKUs within a cluster therefore remain in the same (arbitrary) sequence as it is expected that the sequencing of SKUs within a cluster does not significantly effect the solution quality. The RSC is a basic sequencing approach and relies mainly on the effectiveness of the clustering to generate good solutions. Each of the generated instances is solved with the G4 variation of Algorithm 3 and the best solution based on this measure is further optimised with the HM variation of Algorithm 4. Algorithm 9 contains the pseudo code for the random clustering approach.

Algorithm 9: Random search using clustered SKUs

Input: A set $\mathcal{Q}$ of clustered SKUs where $|\mathcal{Q}| = m^c$
Output: A set $\mathcal{P}$ of SKUs

$\mathcal{P}_{\text{current}} \leftarrow$ a random solution;
$\mathcal{P} \leftarrow \mathcal{P}_{\text{current}}$;
for *Set number of iterations* **do**
    $\mathcal{P}_{\text{current}} \leftarrow$ a random solution;
    if $C_{\text{current}} < C_{\text{best}}$ **then**
        $\mathcal{P} \leftarrow \mathcal{P}_{\text{current}}$
    end
end

5.4.2 Tabu search

Tabu search was introduced by Fred Glover in 1986 [13]. It incorporates "intelligence" which distinguishes it from a local search. In most cases a tabu search is applied to problems with the objective of optimising some objective function $f(\mathbf{z})$ while satisfying some constraints. These constraints define a feasible solution space $\mathcal{Z}$ in which the vector of decision variables $\mathbf{z}$ must lie.

A tabu search is an iterative search where one candidate solution $\mathbf{z}$ moves to another $\mathbf{z}'$. The next candidate solution $\mathbf{z}'$ is said to be in the neighbourhood of the previous solution or $\mathbf{z}' \in \mathcal{N}(\mathbf{z})$ where $\mathcal{N}(\mathbf{z})$ defines a set of candidate solutions adjacent to $\mathbf{z}$. The definition of $\mathcal{N}(\mathbf{z})$ differs between problems, and attempts to take specific problem characteristics into account.

Having a definition for $\mathcal{N}(\mathbf{z})$ with two main characteristics, namely connectivity and reversibility, allows for the tabu search to use "intelligence" to move out of local optima. Connectivity implies that all feasible solutions may be reached by all other feasibly solutions by applying a sequence of moves. Reversibility implies that if a move ψ is applied ($\otimes$) to a solution then a move exists (ψ^{-1}) such that $(\mathbf{z} \otimes \psi) \otimes \psi^{-1} = \mathbf{z}$.

The "intelligence" of the tabu search is realised by prohibiting an inverse move to be made for at least t iterations after the original move is made. This prevents cycles of moves of length t in an effort to move away from local optima. Algorithm 10 contains the pseudo code for the SLP tabu search implementation.

Based on the assumption that the sequence of individual SKUs within a cluster has a negligible effect on solution quality a Tabu search (TC) was implemented which sequences the different clusters. A move is defined as switching the positions of two clusters. The neighbourhood is

Algorithm 10: Tabu search using clustered SKUs

Input: A set $\mathcal{Q}$ of clustered SKUs where $|\mathcal{Q}| = m^c$
Output: A set $\mathcal{P}$ of SKUs

```
P_current ← a random solution;
P ← P_current;
while Stopping criteria not met do
    Generate all neighbours and select the best one, P_neighbour, based on the maximal cut, C, calculated by
    means of the G3 algorithm;
    P_current ← P_neighbour;
    if C_current < C_best then
        P ← P_current
    end
end
```

all solutions within one non-prohibited move of the current solution. The number of iterations for which an inverse move is prohibited is taken as 10% of the number of orders. To determine the quality of a neighbour the G4 variation of Algorithm 3 was used with the final solution undergoing further optimisation with the HM method. A maximum number of iterations was used as stopping criteria. Algorithm 10 contains the pseudo code for this tabu search approach.

5.4.3 Ant colony approach to sequencing clusters

The ant colony algorithm, discussed in §5.3 was adapted to sequence the different clusters. This adapted ant colony approach (ACC) is described in Algorithm 11. It ignores the effect of visibility as no measure which could be applied over all clustering variations could be found and only uses pheromone levels as a guide for the search. Pheromones indicate the desirability of two clusters to be placed adjacent to each other. This algorithm also relies on the assumption that the individual sequence within each cluster does not significantly effect the solution quality. The algorithm may also be viewed as a guided random search as new solutions are generated based on skewed probability distributions. The random transition rule that is used is given by

$$\hat{p}_{ij}^{k}(t) = \begin{cases} \dfrac{(\tau_{ij}(t))^{\alpha}}{\sum_{q_l \in D_i^k}(\tau_{il}(t))^{\alpha}} & \text{if } q_j \in \mathcal{U}_i^k \\ 0 & \text{otherwise,} \end{cases} \tag{5.20}$$

where $\hat{p}_{ij}^{k}(t)$ is the probability that ant k places cluster q_j adjacent to cluster q_i at iteration t. The set $\mathcal{U}_i^k$ is the set of unassigned clusters at the current assignment step. All the parameters were determined by solving benchmark instances with different parameter configurations. The results for the parameter testing are presented in Appendix C.2.

5.5 Data analysis

Twenty two real life data sets were obtained from Pep to test the algorithms presented here. These 22 data sets were then used to generate arbitrary cases where duplicated SKUs are present by duplicating the top 10 most frequently picked SKUs. A simple lower bound in terms of cycles traversed (LBC) for each instance was calculated as

$$\text{LBC} = \max_t \left[\frac{\mathcal{R}(\varsigma_t)}{|\bigcap_{\varsigma_r \leftrightarrow \varsigma_t} \varsigma_r|} \right]. \tag{5.21}$$

Algorithm 11: Ant colony algorithm for clustered SKUs

Input: A set $\mathcal{Q}$ of clustered SKUs where $|\mathcal{Q}| = m^c$
Output: A set $\mathcal{P}$ of SKUs

$C_{\text{best}} = n$;
for $t <$ *iterations* **do**
  for *Each ant* k **do**
    Generate $\mathcal{P}^k(t)$ from the clustered SKUs in sets $\mathcal{Q}$ and solve with greedy heuristic;
    if $C^k(t) < C_{\text{best}}$ **then**
      $C_{\text{best}} \leftarrow C^k(t)$;
      $\mathcal{P} \leftarrow \mathcal{P}^k(t)$;
    end
  end
  Update pheromone levels;
end

This lower bound may be seen as the maximum, over all SKUs, of the number of orders requiring that SKU divided by the number of locations assigned to that SKU. These data sets were used to determine differences in performance between different parameter configurations and between the different methods.

After testing the different algorithms and variations the Bonferoni statistical tests were used to determine if significant differences between algorithms exist. The Bonferoni test is a pairwise multiple comparison test between different sample means. Each comparison is done with a critical value of $2\alpha/k(k-1)$ with the hypothesis

$$H_0 : \mu_i = \mu_j$$
$$H_1 : \mu_i \neq \mu_j$$

for all combinations of samples i and j. If H_0 cannot be rejected the samples will be considered as not significantly different and placed in the same group. Due to the pairwise comparisons the analysis may show that sample i is not significantly different to both samples j and k while samples j and k are significantly different to each other. Therefore samples may be assigned to multiple groups where samples in that group are not significantly different.

To eliminate the effects of different sized instances the data sets were categorised into different classes. Three classes were considered, those with lower bounds greater than 800 (A–J) were classified as Large, those with lower bounds between 800 and 50 (K–Q) as Medium and the remainder as Small (R–V). By splitting instances between classes the effect of vastly different instance sizes is removed. There is, however, still differences in instance sizes within classes. In an attempt to eliminate these differences the results for each data set was normalised by dividing it by the best possible solution obtained by an algorithm under comparison. This is done in an attempt to assign equal weighting to the different data sets in each class. The normalised data may be seen as a collection of samples from a population and one can determine significant differences between the respective means. All statistical analysis were performed with a 95% confidence level and was programmed using SAS 9.1 [29].

5.6 Results

All of the solution approaches and variations thereof where tested to determine the best method. The clustering approaches were tested independently to first determine the best clustering variation and solution method. The best clustering approaches were then compared to the other heuristic and metaheuristic approaches.

5.6.1 Heuristic approaches

All the data sets were solved with all the appropriate heuristics. The data sets with no duplicated SKUs were not solved with the OPA and GPA heuristics described in §5.2 and the data sets with duplicated SKUs were not solved with the GP heuristic described in Algorithm 6.

Class	Data set	Size (n, SKUs, m)	OP	OPA	GP	GPA	Lower Bound
Large	A	(1262, 49, 49)	1233	–	1232*	–	1232
	B	(1264, 54, 54)	1226*	–	1229	–	1226
	C	(1265, 51, 51)	1167	–	1161*	–	1161
	D	(1263, 56, 56)	1073	–	1017*	–	1011
	E	(1264, 51, 51)	1070*	–	1072	–	1069
	F	(1258, 53, 53)	1031	–	1015*	–	959
	G	(1260, 56, 56)	985*	–	985*	–	855
	H	(1244, 54, 54)	1002	–	990*	–	817
	I	(1264, 56, 56)	982	–	967*	–	729
	J	(1258, 55, 55)	944	–	879*	–	835
Medium	K	(943, 63, 63)	291*	–	293	–	95
	L	(846, 56, 56)	224*	–	234	–	141
	M	(728, 51, 51)	153*	–	153*	–	109
	N	(396, 63, 63)	124	–	82*	–	74
	O	(733, 55, 55)	119*	–	122	–	66
	P	(242, 64, 64)	69	–	57*	–	33
	Q	(574, 48, 48)	75*	–	83	–	67
Small	R	(90, 48, 48)	7*	–	7*	–	7
	S	(158, 55, 55)	16*	–	18	–	13
	T	(82, 51, 51)	8*	–	8*	–	8
	U	(80, 56, 56)	6*	–	7	–	5
	V	(89, 42, 42)	10*	–	10*	–	9

Table 5.3: *The solution quality (in number of cycles traversed) obtained by solving all the data sets where no duplicated SKUs are present with the Organ pipe (OP) and Greedy (GP) heuristics and their adaptations (OPA and GPA) described in Algorithms 5 and 6. All heuristics were tested for their appropriate data sets. A dash indicates that the heuristic was not used for that data set and an asterisk indicates the best solution obtained by one of the heuristics. An upper bound may be seen as the number of orders n.*

The results presented in Table 5.3 indicate that for the cases where no SKUs are duplicated, the GP heuristic described in Algorithm 6 outperforms the the OP heuristic described in Algorithm 5 for the large sized data sets. Both the GP and OP perform equally well for the medium and small sized data sets. This suggests that overall the GP outperforms the OP for cases where duplicated SKUs are not present.

For the cases where duplicated SKUs are present the results in Table 5.4 suggest that the OPA heuristic outperforms both the OP and GPA heuristics. For both the large and medium sized data sets the OPA shows the best performance with all heuristics exhibiting similar performances for the small sized data sets.

Based on the results presented in Tables 5.3 and 5.4, for the remainder of the chapter only the GP and OPA heuristics will be considered for further comparison to other SLP approaches.

Class	Data set	Size (n, SKUs, m)	OP	OPA	GP	GPA	Lower Bound
Large	A′	(1262, 49, 59)	1024	914*	–	938	752
	B′	(1264, 54, 64)	957	920*	–	946	633
	C′	(1265, 51, 61)	915	863*	–	892	592
	D′	(1263, 56, 66)	926	873*	–	925	627
	E′	(1264, 51, 61)	914	858*	–	900	534
	F′	(1258, 53, 63)	959	874*	–	908	668
	G′	(1260, 56, 66)	950	871*	–	912	523
	H′	(1244, 54, 64)	964	933*	–	961	602
	I′	(1264, 56, 66)	932	863*	–	914	459
	J′	(1258, 55, 65)	810	732*	–	793	687
Medium	K′	(943, 63, 73)	278	239*	–	275	69
	L′	(846, 56, 66)	206	163*	–	201	70
	M′	(728, 51, 61)	126	111*	–	117	57
	N′	(396, 63, 73)	101	105	–	70*	37
	O′	(733, 55, 65)	107	91*	–	104	33
	P′	(242, 64, 74)	66	62	–	56*	19
	Q′	(574, 48, 58)	56	50*	–	53	33
Small	R′	(90, 48, 58)	4*	4*	–	4*	3.5
	S′	(158, 55, 65)	13	11*	–	13	6
	T′	(82, 51, 61)	5*	7	–	6	4
	U′	(80, 56, 66)	5*	5*	–	5*	3
	V′	(89, 42, 52)	7	7	–	6*	4

Table 5.4: *The solution quality (in number of cycles traversed) obtained by solving all the data sets where duplicated SKUs are present with all the Organ pipe (OP) and Greedy (GP) heuristics and their adaptations (OPA and GPA) described in Algorithms 5 and 6. All heuristics were tested for their appropriate data sets. A dash indicates that the heuristic was not used for that data set and an asterisk indicates the best solution obtained by one of the heuristics. An upper bound may be seen as the number of orders* n.

5.6.2 Clustering variations

To test the performance of each clustering variation (MA, AD, SAD, SA) of Algorithm 8 the RSC approach described in Algorithm 9 was used for each clustering variation to solve the SLP. Two additional parameters were tested for each variation namely the maximum size of a cluster and the number of clusters. The best configuration was determined for each variation. The results of the parameter testing is presented in Appendix D. The clustering variations were then compared to each other by using the best combination of parameters for each clustering variation.

From the results in Table 5.5 it is suggested that there is no significant difference, in terms of cycles traversed, between the SA, MA and SAD clustering variations. They are all placed in the same group (A) for large and small sized data sets where duplicated SKUs are not present. From the results in Table 5.8 it follows that the SAD clustering variation performs better for the large sized data sets as in most cases the best solution was obtained using this clustering variation. It is further suggested from the results in Table 5.6 that there is no significant difference between any of the clustering variations for the medium sized data sets.

The SA clustering variation shows the best mean score for the large sized data sets with no duplicates and the SAD clustering variation for the medium and small sized data sets. Based on this result as well as the number of best solutions obtained by the SAD clustering variation for large sized data sets these two variations will be used when solving SLP instances in these

Class	Data set	Size (n, SKUs, m)	MA	AD	SAD	SA	Lower Bound
Large	A	(1262, 49, 49)	1232*	1236	1232*	1232*	1232
	B	(1264, 54, 54)	1226*	1226*	1226*	1226*	1226
	C	(1265, 51, 51)	1161*	1162	1161*	1161*	1161
	D	(1263, 56, 56)	1014	1040	1012*	1015	1011
	E	(1264, 51, 51)	1070	1072	1069*	1069*	1069
	F	(1258, 53, 53)	1001	1014	992*	1004	959
	G	(1260, 56, 56)	943*	968	944	955	855
	H	(1244, 54, 54)	959	994	958*	979	817
	I	(1264, 56, 56)	940	953	936*	954	729
	J	(1258, 55, 55)	884	952	878	874*	835
Medium	K	(943, 63, 63)	246	263	234	232*	95
	L	(846, 56, 56)	227	228	226*	232	141
	M	(728, 51, 51)	152	146*	156	149	109
	N	(396, 63, 63)	78	95	77	76*	74
	O	(733, 55, 55)	122	119	108*	111	66
	P	(242, 64, 64)	37	44	35	33*	33
	Q	(574, 48, 48)	85	79*	84	83	67
Small	R	(90, 48, 48)	7*	7*	7*	7*	7
	S	(158, 55, 55)	15	14	14	13*	13
	T	(82, 51, 51)	8*	9	8*	8*	8
	U	(80, 56, 56)	5*	6	5*	5*	5
	V	(89, 42, 42)	9*	10	9*	9*	9

Table 5.5: *The solution quality (in number of cycles traversed) obtained by solving all the data sets where no duplicated SKUs are present using the Maximum Adjacencies (MA), Adjacency Domination (AD), SKU Adjacency Domination (SAD) and SKU Adjacency (SA) clustering variations of Algorithm 8. Each variation was solved with its best parameter configuration and the clusters were sequenced with the RSC cluster sequencing approach. The best solution obtained is indicated by an asterisk. An upper bound may be seen as the number of orders n.*

Class	Bonferoni group	Mean	Cluster Variation
Large	A	1.003724	SAD
	A,B	1.006044	MA
	A,B	1.010083	SA
	B	1.02612	AD
Medium	A	1.05166	SA
	A	1.06476	SAD
	A	1.10006	MA
	A	1.15246	AD
Small	A	1	SA
	A,B	1.01538	SAD
	A,B	1.03077	MA
	B	1.10261	AD

Table 5.6: *The Bonferoni groupings and mean scores (solution quality relative to the best solution obtained) for all classes of data where duplicated SKUs are not present for the Maximum Adjacencies (MA), Adjacency Domination (AD), SKU Adjacency Domination (SAD) and SKU Adjacency (SA) clustering variations of Algorithm 8. Each variation was solved with its best parameter configuration and the clusters were sequenced with the RSC cluster sequencing approach. Elements with the same group within the same class exhibit no significant difference in performance.*

respective classes.

Class	Data set	Size (n, SKUs, m)	MA	AD	SAD	SA	Lower Bound
Large	A′	(1262, 49, 59)	900	896*	922	924	752
	B′	(1264, 54, 64)	879*	904	883	884	633
	C′	(1265, 51, 61)	813	837	823	807*	592
	D′	(1263, 56, 66)	873*	874	877	888	627
	E′	(1264, 51, 61)	844*	852	864	844*	534
	F′	(1258, 53, 63)	868	862	836*	893	668
	G′	(1260, 56, 66)	822*	861	840	847	523
	H′	(1244, 54, 64)	898*	932	920	919	602
	I′	(1264, 56, 66)	847	867	841*	849	459
	J′	(1258, 55, 65)	742	728*	733	729	687
Medium	K′	(943, 63, 73)	206	227	191*	197	69
	L′	(846, 56, 66)	163	166	154*	161	70
	M′	(728, 51, 61)	100	102	98*	107	57
	N′	(396, 63, 73)	49	69	47*	49	37
	O′	(733, 55, 65)	92	91	79	78*	33
	P′	(242, 64, 74)	31	51	25*	28	19
	Q′	(574, 48, 58)	51	47	46*	47	33
Small	R′	(90, 48, 58)	4*	4*	4*	4*	3.5
	S′	(158, 55, 65)	9	11	7*	8	6
	T′	(82, 51, 61)	5	5	4*	4*	4
	U′	(80, 56, 66)	4*	5	4*	4*	3
	V′	(89, 42, 52)	5*	5*	5*	5*	4

Table 5.7: *The solution quality (in number of cycles traversed) obtained by solving all the data sets where duplicated SKUs are present using the Maximum Adjacencies (MA), Adjacency Domination (AD), SKU Adjacency Domination (SAD) and SKU Adjacency (SA) clustering variations of Algorithm 8. Each variation was solved with its best parameter configuration and the clusters were sequenced with the RSC cluster sequencing approach. The best solution obtained is indicated by an asterisk. An upper bound may be seen as the number of orders n.*

It is suggested from the results presented in Table 5.8 that there is no significant difference between the clustering variations of Algorithm 8 for both the large and small sized data sets where duplicated SKUs are present. The results in Table 5.7 suggest marginally better performances for the MA clustering variation as in most cases the best solution was obtained by this variation. Due to the higher mean score and the results in Table 5.8 the MA clustering variation will be used when solving the SLP for large sized data sets where duplicated SKUs are present.

Some significant differences between variations exist for medium sized data sets with duplicated SKUs with the SAD clustering variation having the best mean score shown in Table 5.8. This hypothesis is strengthened by the number of times the best solution was obtained by the SAD clustering variation for medium sized data sets shown in Table 5.7. The SAD clustering variation will therefore be used for medium and small sized data sets where duplicated SKUs are present.

The algorithm with the best performance (without duplication) differ from those for the cases where duplicated SKUs are not present. For the cases where duplicated SKUs are present the MA clustering variation of Algorithm 8 shows the best mean score in contrast to the SA clustering variation for large sized data sets and for medium and small sized data sets. The SAD clustering variation shows a slightly better mean score than the SA clustering variation.

Finally, it may be concluded that the AD clustering variation of Algorithm 8 has the worst performance for all types of data. This is evident from the fact that AD clustering variation

Class	Bonferoni group	Mean	Cluster Variation
Large	A	1.016873	MA
	A	1.022706	SAD
	A	1.027977	SA
	A	1.031629	AD
Medium	A	1.03653	SAD
	A,B	1.08659	SA
	A,B	1.14121	MA
	B	1.32629	AD
Small	A	1	SAD
	A	1.02857	SA
	A	1.10714	MA
	A	1.21429	AD

Table 5.8: *The Bonferoni groupings and mean scores (solution quality relative to the best solution obtained) for all classes of data where duplicated SKUs are present for the Maximum Adjacencies (MA), Adjacency Domination (AD), SKU Adjacency Domination (SAD) and SKU Adjacency (SA) clustering variations of Algorithm 8. Each variation was solved with its best parameter configuration and the clusters were sequenced with the RSC cluster sequencing approach. Elements with the same group within the same class exhibit no significant difference in performance.*

shows the worst mean score for all data classes. For the remainder of the thesis when using a clustering algorithm to solve a SLP instance the best variation for the instance's class will be used.

5.6.3 Comparison of cluster sequencing approaches

To compare the three sequencing approaches the best clustering variation was used for each data class. For the RSC cluster sequencing approach (described in Algorithm 9) and ACC cluster sequencing approach (described in Algorithm 11) the best parameter configurations were used for each variation. Both the RSC and ACC cluster sequencing approaches were configured such that each would generate the same number of candidate solutions and therefore solve the same number of OSP instances. The TC cluster sequencing approaches (described in Algorithm 10) generates more solutions which resulted in greater computational times during preliminary testing. When testing clustering variation parameters (number of clusters and maximum cluster size) in some cases no significant differences were found between parameter combinations. Therefore to reduce the TC neighbourhood size, and thus the number of OSP instances solved, the number of clusters was set to a minimum within the best Bonferoni grouping of parameter configurations. This resulted in setting the number of clusters to 6 for the TC cluster sequencing approach. The parameters for the ACC cluster sequencing approach were determined by testing different parameter configurations. The results of the parameter testing is Appendix D

The results presented in Table 5.10 suggests that the RSC and TC cluster sequencing approaches show no significant difference when applied to large and medium sized data sets. The ACC cluster sequencing approach displays the worst performance for large sized data sets and no significant difference with the TC cluster sequencing approach for medium sized data sets. This suggest that the RSC cluster sequencing approach is the best sequencing approach for medium sized data sets. There is no significant difference between sequencing approaches for the small sized data sets. Based on the results in Table 5.9 and Table 5.10 the RSC sequencing approach

Class	Data set	Size (n, SKUs, m)	RSC	TC	ACC	Lower Bound
Large	A	(1262, 49, 49)	1232*	1232*	1241	1232
	B	(1264, 54, 54)	1226*	1226*	1226*	1226
	C	(1265, 51, 51)	1161*	1161*	1161*	1161
	D	(1263, 56, 56)	1012*	1021	1020	1011
	E	(1264, 51, 51)	1069*	1069*	1069*	1069
	F	(1258, 53, 53)	992*	994	1011	959
	G	(1260, 56, 56)	944*	945	949	855
	H	(1244, 54, 54)	958*	963	969	817
	I	(1264, 56, 56)	936	934*	941	729
	J	(1258, 55, 55)	878	877*	888	835
Medium	K	(943, 63, 63)	232	230*	232	95
	L	(846, 56, 56)	232	230*	232	141
	M	(728, 51, 51)	149*	152	151	109
	N	(396, 63, 63)	76*	76*	78	74
	O	(733, 55, 55)	111	111	110*	66
	P	(242, 64, 64)	33*	34	36	33
	Q	(574, 48, 48)	83	83	79*	67
Small	R	(90, 48, 48)	7*	7*	7*	7
	S	(158, 55, 55)	13*	13*	13*	13
	T	(82, 51, 51)	8*	8*	8*	8
	U	(80, 56, 56)	5*	6	6	5
	V	(89, 42, 42)	9*	9*	9*	9

Table 5.9: *The solution quality (in number of cycles traversed) obtained by solving all the data sets where duplicated SKUs are present with the random (RSC), tabu search (TC) and ant colony adaptation (ACC) cluster sequencing approaches. The best clustering variations were used with their best parameters except in the case where the TC was used where the number of clusters was 6 and the best cluster size was used. The best solution obtained is indicated by an asterisk. An upper bound may be seen as the number of orders n.*

Class	Bonferoni group	Mean	Sequencing Approach
Large	A	1.000328	RSC
	A	1.001719	TC
	B	1.007118	ACC
Medium	A	1.01102	RSC
	A, B	1.01574	TC
	B	1.02115	ACC
Small	A	1	RSC
	A	1.04	ACC
	A	1.04	TC

Table 5.10: *The Bonferoni groupings and mean scores (solution quality relative to the best solution obtained) of the random (RSC), tabu search (TC) and ant colony adaptation (ACC) cluster sequencing approaches for all classes of data where duplicated SKUs are not present. Elements with the same group within the same class exhibit no significant difference in performance.*

will be used for instances where no duplicated SKUs are present.

The poorer performance for the ACC cluster sequencing approach may be attributed to poor parameter selection and the requirement that the number of generated candidate solutions should be equal to that of the RSC.

Class	Data set	Size (n, SKUs, m)	RSC	TC	ACC	Lower Bound
Large	A′	(1262, 49, 59)	900*	1024	914	752
	B′	(1264, 54, 64)	879*	957	920	633
	C′	(1265, 51, 61)	813*	873	863	592
	D′	(1263, 56, 66)	873*	876	873*	627
	E′	(1264, 51, 61)	844*	858	858	534
	F′	(1258, 53, 63)	868*	926	874	668
	G′	(1260, 56, 66)	822*	852	871	523
	H′	(1244, 54, 64)	898*	921	933	602
	I′	(1264, 56, 66)	847*	893	863	459
	J′	(1258, 55, 65)	742	721*	732	687
Medium	K′	(943, 63, 73)	191	187*	239	69
	L′	(846, 56, 66)	154*	155	163	70
	M′	(728, 51, 61)	98	97*	111	57
	N′	(396, 63, 73)	47	48	105	37
	O′	(733, 55, 65)	79*	81	91	33
	P′	(242, 64, 74)	25*	25*	62	19
	Q′	(574, 48, 58)	46	44*	50	33
Small	R′	(90, 48, 58)	4*	4*	4*	3.5
	S′	(158, 55, 65)	7*	8	11	6
	T′	(82, 51, 61)	4*	4*	7	4
	U′	(80, 56, 66)	4	3*	5	3
	V′	(89, 42, 52)	5*	5*	7	4

Table 5.11: *The solution quality (in number of cycles traversed) obtained by solving all the data sets where duplicated SKUs are present with the random (RSC), tabu search (TC) and ant colony adaptation (ACC) cluster sequencing approaches. The best clustering variations were used with their best parameters except in the case where the TC was used where the number of clusters was 6 and the best cluster size was used. The best solution obtained is indicated by an asterisk. An upper bound may be seen as the number of orders n.*

Class	Bonferoni group	Mean	Sequencing Approach
Large	A	1.00291	RSC
	B	1.02972	ACC
	B	1.03786	TC
Medium	A	1.01076	TC
	A	1.01413	RSC
	B	1.06012	ACC
Small	A	1.01076	TC
	A	1.01413	RSC
	B	1.06012	ACC

Table 5.12: *The Bonferoni groupings and mean scores (solution quality relative to the best solution obtained) of the random (RSC), tabu search (TC) and ant colony adaptation (ACC) cluster sequencing approaches for all classes of data where duplicated SKUs are present. Elements with the same group within the same class exhibit no significant difference in performance.*

From the results presented in Table 5.12 it may be concluded that the RSC cluster sequencing approach performs the best for large sized data sets. The same conclusion may be reached from the results presented in Table 5.11 as the solutions obtained by using this approach were the best for most instances. Again the poorer performances of the ACC cluster sequencing approach may be attributed to poor parameter selection and reduced number of generated solutions to equal the RSC cluster sequencing approach. It may be noted that the performance of the TC cluster sequencing approach has reduced dramatically for the large sized data sets when duplicated SKUs are present. The RSC cluster sequencing approach will therefore be used for large sized data sets where duplicated SKUs are present.

For medium and small sized data sets it is suggested from the results presented in Table 5.12 that the RSC cluster sequencing approach and the TC cluster sequencing approach show no significant differences. The TC cluster sequencing approach displays a better mean score in contrast to the data sets where duplicated SKUs are not present. These results should, however, be viewed with the computational times in Table 5.13 before considering the TC cluster sequencing approach.

Class	TC	RSC	ACC
Large	(336.0, 4.57)	(109.9, 3.36)	(111.4, 3.21)
Medium	(95.7, 6.57)	(19.2, 3.23)	(19.9, 3.14)
Small	(12.1, 2.05)	(1.6, 0.82)	(1.9, 0.87)
Large*	(389.4, 5.1)	(137, 4.41)	(139.1, 3.65)
Medium*	(104.8, 6.96)	(22.9, 3.54)	(22.9, 3.38)
Small*	(12.5, 2.09)	(1.8, 0.88)	(2.2, 0.93)

Table 5.13: *The average computational times for the random (RSC), tabu search (TC) and ant colony adaptation (ACC) cluster sequencing approaches for each data class. The result are given in seconds with each duple representing the average and standard deviation (μ, σ) over all the data sets in the class.*

It may be concluded from the a summary of the computational times presented in Table 5.13 that the TC cluster sequencing approach has the longest computational times with the RSC and ACC cluster sequencing approaches showing similar computational times. The similar computational times of the RSC and ACC cluster sequencing approaches is attributed to the same number of solutions generated. The longer computational times realised by the TC cluster sequencing approach may be explained by the number of solutions generated in a single neighbourhood which must all be evaluated with the G4 greedy variation of Algorithm 3. The computational times for the TC cluster sequencing approach is on average at least 4 times greater than the RSC cluster sequencing approach for medium and small sized data sets. Although the TC cluster sequencing approach investigated a larger part of the solution space the neighbourhood was too local and the increase in computational time was not compensated for by the small increase in solution quality. Therefore the RSC cluster sequencing approach will be used for large as well as medium and small sized data sets.

5.6.4 All approaches

To determine the best algorithm for solving the SLP the best heuristic and metaheuristic algorithms were compared. The best heuristic approaches discussed in §5.6.1, the best clustering variation and cluster sequencing approach discussed in §5.6.2 and §5.6.3, the ant colony approach (ASA) described in Algorithm 7 and a random approach RA are all compared.

The results presented in Table 5.14 suggest that no significant difference exists between any two algorithms. The results in Table 5.15 are summarised in Figure 5.2 and suggest that the RSC cluster sequencing approach described in Algorithm 9 with an appropriate clustering variation has the best performance for large sized data sets as in all cases the best performance was shown by this algorithm. In addition, for all data sizes the RSC cluster sequencing approach showed the best mean score. The RSC cluster sequencing approach with an appropriate clustering variation

Class	Bonferoni group	Mean	Algorithm
Large	A	1.00197	RSC
	A	1.00686	ASA
	A	1.01642	GP
	A	1.03564	RA
Medium	A	1.01366	RSC
	A	1.02942	ASA
	A	1.12921	RA
	A	1.18608	GP
Small	A	1	RSC
	A	1.05538	RA
	A	1.07077	ASA
	A	1.17915	GP

Table 5.14: *The Bonferoni groupings and mean scores (solution quality relative to the best solution obtained) of the random approach (RA), a greedy heuristic (GP), ant colony approach (ASA) and the random cluster sequencing approach with an appropriate clustering variation (RSC) for all classes of data where no duplicated SKUs are present. Elements with the same group within the same class exhibit no significant difference in performance.*

Class	Data set	Size (n, SKUs, m)	RA	GP	ASA	RSC	Lower Bound
Large	A	(1262, 49, 49)	1232*	1232*	1233	1232*	1232
	B	(1264, 54, 54)	1226*	1229	1226*	1226*	1226
	C	(1265, 51, 51)	1161*	1161*	1161*	1161*	1161
	D	(1263, 56, 56)	1075	1017	1026	1012*	1011
	E	(1264, 51, 51)	1069*	1072	1069*	1069*	1069
	F	(1258, 53, 53)	1012	1015	1014	992*	959
	G	(1260, 56, 56)	963	985	958	944*	855
	H	(1244, 54, 54)	998	990	962	958*	817
	I	(1264, 56, 56)	953	967	948	936*	729
	J	(1258, 55, 55)	1028	879	861*	878	835
Medium	K	(943, 63, 63)	263	293	240	232*	95
	L	(846, 56, 56)	232	234	227*	232	141
	M	(728, 51, 51)	152	153	161	149*	109
	N	(396, 63, 63)	90	82	74*	76	74
	O	(733, 55, 55)	125	122	110*	111	66
	P	(242, 64, 64)	45	57	36	33*	33
	Q	(574, 48, 48)	81	83	80*	83	67
Small	R	(90, 48, 48)	7*	7*	7*	7*	7
	S	(158, 55, 55)	14	18	15	13*	13
	T	(82, 51, 51)	8*	8*	8*	8*	8
	U	(80, 56, 56)	6	7	6	5*	5
	V	(89, 42, 42)	9*	10	9*	9*	9

Table 5.15: *The solution quality (in number of cycles traversed) obtained by solving all the data sets where duplicated SKUs are not present for the random approach (RA), a greedy heuristic (GP), ant colony approach (ASA) and the random cluster sequencing approach with an appropriate clustering variation (RSC). The best solution obtained by one of the algorithms is indicated by an asterisk. An upper bound may be seen as the number of orders n.*

will therefore be used when solving SLP instances where duplicated SKUs are not present.

It should be noted that the GP heuristic described in Algorithm 6 shows the worst mean score

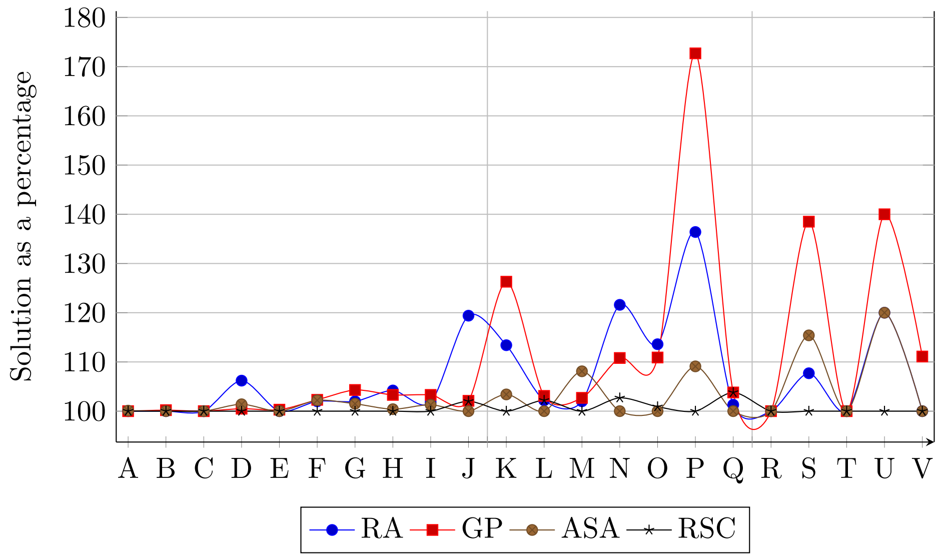


Figure 5.2: *A graphical illustration of the comparison between the random approach (RA), a greedy heuristic (GP), ant colony approach (ASA), the random cluster sequencing approach with an appropriate clustering variation (RSC) and the best solution obtained by any one of the approaches for historical data sets. The best solution obtained by one of the approach represents the 100% benchmark.*

for medium and small sized data sets and second worst score for large sized data sets. This suggests that the heuristics, although proven to be optimal for carousel systems with stochastic orders, do not perform well when a finite set of deterministic orders are known [16].

Class	Bonferoni group	Mean	Algorithm
Large	A	1.006881	RSC
	A,B	1.025854	ASA
	B	1.032009	OPA
	C	1.074878	RA
Medium	A	1	RSC
	A	1.1019	ASA
	A	1.4336	RA
	A	1.485	OPA
Small	A	1	RSC
	A	1.0571	ASA
	A	1.3614	RA
	A	1.3943	OPA

Table 5.16: *The Bonferoni groupings and mean scores (solution quality relative to the best solution obtained) of the random approach (RA), a greedy heuristic (GP), ant colony approach (ASA) and the random cluster sequencing approach with an appropriate clustering variation (RSC) for all classes of data where duplicated SKUs are present. Elements with the same group within the same class exhibit no significant difference in performance.*

The results presented in Table 5.16 suggest that there is no significant difference between the RSC cluster sequencing approach and the ASA approach described in Algorithm 7 for any data size. However, the RSC cluster sequencing approach has the best mean score and the best solution was obtained for all data sets as shown in Table 5.17 and are summarised in Figure 5.3. It may therefore be concluded that the RSC cluster sequencing approach with an appropriate clustering variation is the best algorithm for all instances.

Class	Data set	Size (n, SKUs, m)	RA	OPA	ASA	RSC	Lower Bound
Large	A′	(1262, 49, 49)	935	914	917	900*	752
	B′	(1264, 54, 54)	941	920	912	879*	633
	C′	(1265, 51, 51)	888	863	854	813*	592
	D′	(1263, 56, 56)	922	873*	896	873*	627
	E′	(1264, 51, 51)	898	858	880	844*	534
	F′	(1258, 53, 53)	905	874	827*	868	668
	G′	(1260, 56, 56)	908	871	849	822*	523
	H′	(1244, 54, 54)	956	933	912	898*	602
	I′	(1264, 56, 56)	911	863	876	847*	459
	J′	(1258, 55, 55)	792	732	728*	742	687
Medium	K′	(943, 63, 63)	273	239	210	191*	69
	L′	(846, 56, 56)	198	163	174	154*	70
	M′	(728, 51, 51)	116	111	103	98*	57
	N′	(396, 63, 63)	70	105	49	47*	37
	O′	(733, 55, 55)	104	91	86	79*	33
	P′	(242, 64, 64)	55	62	32	25*	19
	Q′	(574, 48, 48)	52	50	47	46*	33
Small	R′	(90, 48, 48)	4*	4*	4*	4*	3.5
	S′	(158, 55, 55)	13	11	9	7*	6
	T′	(82, 51, 51)	6	7	4*	4*	4
	U′	(80, 56, 56)	5	5	4*	4*	3
	V′	(89, 42, 42)	6	7	5*	5*	4

Table 5.17: *The solution quality (in number of cycles traversed) obtained by solving all the data sets where duplicated SKUs are present for the random approach (RA), a greedy heuristic (GP), ant colony approach (ASA) and the random cluster sequencing approach with an appropriate clustering variation (RSC). The best solution obtained by one of the algorithms is indicated by an asterisk. An upper bound may be seen as the number of orders n.*

It should be noted that the RA approach performs significantly worse than the other algorithms for large sized data sets. This is in contrast to the results where duplicated SKUs are not present. These results suggest that "intelligence" is required when assigning the duplicated SKUs to locations. Although the OPA heuristic described in §5.2 was found to be the best heuristic approach in §5.2 it performs poorly for all data sizes when compared to other solution approaches.

5.7 Chapter Summary

The SLP was modelled as an integer programming problem and the exact formulation was presented in §5.1. The formulation was based on a combination of assignment and TSP problem formulations due to the complexity and size exact solution methods were not used. Heuristic and metaheuristic solutions to the problem were therefore investigated.

Two known heuristics for product locations on carousel systems (OP and GP), were tested and further adapted for cases where duplicated SKUs were present. In addition an ant colony algorithm (ASA) was developed making use of principles applied to TSPs and variants thereof.

In an attempt to reduce the size of the investigated solution space clustering algorithms were developed to cluster different SKUs together and treating them as one entity. Four clustering variations were tested and the best one for different data classes used. Once SKUs were placed into clusters three cluster sequencing approaches were tested to arrange these different clusters,

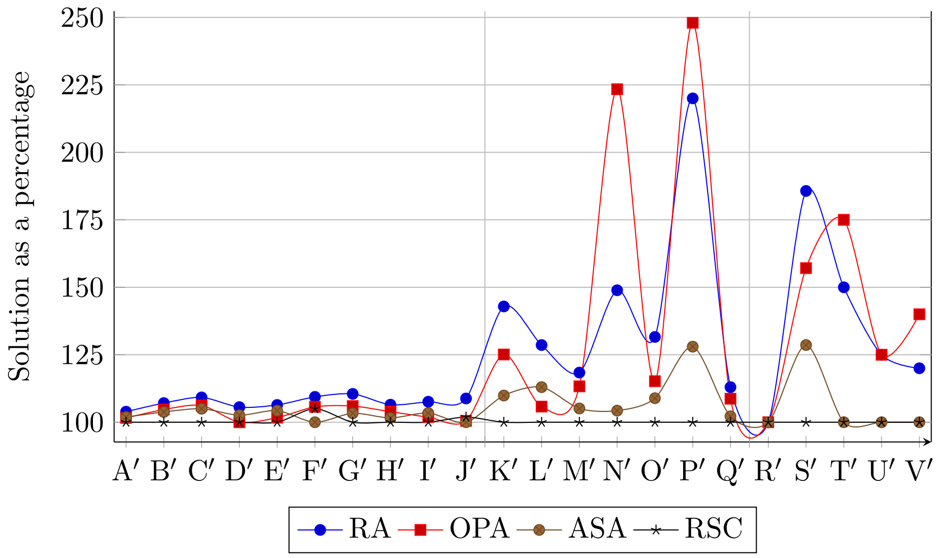


Figure 5.3: *A graphical illustration of the comparison between the random approach (RA), a greedy heuristic (GP), ant colony approach (ASA), the random cluster sequencing approach with an appropriate clustering variation (RSC) and the best solution obtained by one of the approaches for SLP instances with duplicated SKUs. The best solution obtained by any one of the approaches represents the 100% benchmark.*

namely: a random search (RSC), an ant colony variation (ACC) and a Tabu search (TC). It was concluded that the best approach to solving the SLP was to use the best clustering variation in conjunction with the random search cluster sequencing approach.

CHAPTER 6

Constrained SLP

Contents

One of the assumptions made when formulating the SLP in Chapter 5 is that any pair of SKUs may be placed adjacent to each other. Pep's management, however, has a policy where SKUs of the same colour and style and which differ only by size are not allowed to be placed adjacent to each other on a picking line. For example the SKU representing a style of black pants size L may not be placed adjacent to the SKU representing the same style of black pants size M. This additional constraint, which will be referred to as the colour feasibility constraint, is imposed to lower the risk of picking inaccuracies and the accidental mixing of stock on the picking line. Usually when SKUs differ only in size their branch profile, or the set of branches requiring that SKU, are similar and therefore many adjacencies exist between the two SKUs and it is expected that the colour feasibility constraints should be restrictive. Solution approaches for the SLP case with the inclusion of these colour feasibility constraints (SLPCF) is investigated in this Chapter.

To better describe the colour feasibility constraints let the set $\mathcal{X}_t$ be the set of SKUs which may not be adjacent to SKU ς_t, or the set of SKUs similar to SKU ς_t. The colour feasibility constraints may be written, in terms of variables and parameters of Formulation (5.1)–(5.12), as

$$\varsigma_{tj} + \varsigma_{r,j+1} \leq 1 \qquad \forall\ t, r, j, \text{ where } r \in \mathcal{X}_t. \tag{6.1}$$

Equation set (6.1) ensures that there is always at least one location between similar SKUs.

To determine the effect of these constraints the SLPCF was solved making use of similar algorithms to the SLP. Where possible the algorithms used for solving the SLP were adjusted and tested for use on the SLPCF.

6.1 Necessary changes to algorithms

The heuristic methods (OP, GP, OPA, GPA) discussed in §5.2, assigned locations to SKUs on the basis of pick frequency. Incorporating this single attributed measure enabled these heuristics to be fast. Trying to adapt the heuristic methods to satisfy the colour feasibility constraints would change this computational simplicity and therefore the heuristics discussed in §5.2 are not considered for adjustment to solve the SLPCF.

The ASA approach described in Algorithm 7 is adapted in two ways. Firstly if a potential solution broke the colour feasibility constraints the solution quality will be set to the upper bound (number of orders n). Furthermore, the random proportional transition rule (5.13) is changed to

$$p_{ij}^k(t) = \begin{cases} \dfrac{[\tau_{ij}(t)]^{\alpha}\cdot(\nu_{ij})^{\beta}}{\sum_{\varsigma_l\in\mathcal{U}_i^k}[\tau_{il}(t)]^{\alpha}\cdot(\nu_{ij})^{\beta}} & \text{if } \varsigma_j\in\mathcal{U}_i^k \text{ and } \varsigma_j\notin\mathcal{X}_i \\ 0 & \text{otherwise,} \end{cases} \tag{6.2}$$

which reduces the probability of placing two similar SKUs adjacent to each other to 0.

All clustering variations discussed in §5.4 are adjusted in the same way by changing the calculation of the distance measure to

$$d^a(q_i,q_j) = \begin{cases} d(q_i,q_j) & \text{if } \mathcal{C}(q_i,q_j)\leq\left\lfloor\frac{|q_i|+|q_j|}{2}\right\rfloor \\ 0 & \text{otherwise,} \end{cases} \tag{6.3}$$

where $\mathcal{C}(q_i,q_j)$ is the size of the largest set of similar SKUs present in one of the clusters q_i or q_j. Equation (6.3) ensures that within each cluster a configuration may be found which satisfies the colour feasibility constraints. In addition, if the size of the largest set of similar SKUs in the picking line ($\mathcal{C}(\mathcal{Q})=\max_i|\mathcal{X}_i|$) is less than half the number of clusters generated, or $\mathcal{C}(\mathcal{Q})\leq\lfloor m^c/2\rfloor$, a feasible sequence of clusters would always exist. The proof for this result is given in Theorem 5.

Theorem 5. *A feasible solution to the SLPCF exists if $\mathcal{C}(\mathcal{Q})\leq\lfloor m^c/2\rfloor$, using the agglomerative hierarchical clustering algorithm given in Algorithm 8 and distance measure given in equation* (6.3)*.*

Proof. Consider a digraph $\hat{\mathcal{D}}$ and let each vertex $v\in V(\hat{\mathcal{D}})$ represent a cluster generated by the clustering procedure and distance measure given in equation (6.3). Let (u,v) be an arc in $\hat{\mathcal{D}}$ if the cluster associated with v may be placed to the right of the cluster associated with u in the picking line. A Hamiltonian path in $\hat{\mathcal{D}}$ would then represent a feasible solution to the SLPCF. Therefore if $\hat{\mathcal{D}}$ can be shown to be Hamiltonian then a solution exists to the SLPCF.

Let us consider the in-degree of a vertex $d^-(v)$ and the out-degree of a vertex $d^+(v)$ and the number of vertices (clusters) $|V(\hat{\mathcal{D}})|=m^c$. Woodall [35] showed that if $d^-(v)+d^+(u)\geq m^c$ for all vertices where (u,v) is not an arc of $\hat{\mathcal{D}}$ then $\hat{\mathcal{D}}$ is Hamiltonian.

$\mathcal{C}(\mathcal{Q})$ may be seen as the size of largest set of non-adjacent vertices in $\hat{\mathcal{D}}$ which implies that

$$\begin{aligned} d^-(v) &\geq m^c-\mathcal{C}(\mathcal{Q}) && \forall\ v\in V(\hat{\mathcal{D}}),\text{ and}\\ d^+(u) &\geq m^c-\mathcal{C}(\mathcal{Q}) && \forall\ u\in V(\hat{\mathcal{D}}). \end{aligned}$$

and therefore

$$d^-(v) + d^+(u) \geq 2m^c - 2\mathcal{C}(\mathcal{Q}) \qquad \forall\ u, v \in V(\hat{\mathcal{D}}).$$

Furthermore $\mathcal{C}(\mathcal{Q}) \leq \lfloor m^c/2 \rfloor$ implies

$$\begin{aligned} d^-(v) + d^+(u) &\geq 2m^c - 2\lfloor m^c/2 \rfloor && \forall\ u, v \in V(\hat{\mathcal{D}}) \\ d^-(v) + d^+(u) &\geq m^c && \forall\ u, v \in V(\hat{\mathcal{D}}) \end{aligned}$$

Therefore $\hat{\mathcal{D}}$ is Hamiltonian and a feasible solution to the SLPCF exists using the clustering procedure and distance measure given in equation (6.3) if $\mathcal{C}(\mathcal{Q}) \leq \lfloor m^c/2 \rfloor$.

□

6.1.1 Clustering variations

To determine if there are any differences in the performance of clustering configurations using the new distance measure given in equation (6.3) all clustering variations were tested with these new constraints. As in §5.4 the best configuration of the number of clusters and maximum cluster size was used for each clustering variation. The results for the parameter testing are presented in Appendix E. All the data sets were solved with each clustering variation using the RSC cluster sequencing approach described in Algorithm 9 as this approach showed the best results. The data was normalised in the same way as discussed in §5.4 and the Bonferoni test done to determine significant differences in cluster variations.

Class	Bonferoni group	Mean	Cluster Variation
Large	A	1.00395	SAD
	A	1.01748	AD
	A	1.02518	MA
	A	1.02860	SA
Medium	A	1.01818	SAD
	A	1.03372	SA
	A	1.09572	MA
	A	1.12557	AD
Small	A	1.00000	SA
	A	1.01538	SAD
	A	1.05934	MA
	A	1.06410	AD

Table 6.1: *The Bonferoni groupings and mean scores (solution value relative to best solution obtained) of the Maximum Adjacencies (MA), Adjacency Domination (AD), SKU Adjacency Domination (SAD) and SKU Adjacency (SA) clustering variations using the colour feasibility constraints where duplicated SKUs are not present. The best parameter configurations for all classes of data was used for each variation. Elements with the same group within the same class exhibit no significant difference in performance.*

From the results in Table 6.1 it is concluded that there is no significant difference in the performance of the clustering variations for any of the data sizes where duplicated SKUs are not present. The SAD clustering variation shows the best mean score for large and medium sized data sets and the SA clustering variation shows the best mean score for small sized data sets. These two clustering variations will therefore be used to solve instances for their respective sizes.

These results differ from those for the SLP shown in Tables 5.6 where significant differences did exist between different clustering variations and the clustering variation with the best mean score for the medium sized data was the SA clustering variation.

Class	Bonferoni group	Mean	Cluster Variation
Large	A	1.009859	SAD
	A, B	1.017365	SA
	A, B	1.019598	MA
	B	1.037087	AD
Medium	A	1.00997	SAD
	A	1.02938	SA
	A, B	1.16418	MA
	B	1.21626	AD
Small	A	1	SAD
	A	1.025	SA
	A	1.165	MA
	A	1.225	AD

Table 6.2: *The Bonferoni groupings and mean scores (solution value relative to best solution obtained) of the the Maximum Adjacencies (MA), Adjacency Domination (AD), SKU Adjacency Domination (SAD) and SKU Adjacency (SA) clustering variations where duplicated SKUs are present. The best parameter configurations for all classes of data was used for each variation. Elements with the same group within the same class exhibit no significant difference in performance.*

It may be derived from the results in Table 6.2 that significant differences exists in the performance of the clustering variations for the large and medium sized data sets where duplicated SKUs are present. The SAD variation shows the best mean score for all data sizes and will therefore be applied to solve data sets in which duplicated SKUs are present.

6.1.2 Results

Both the ASA and RSC algorithms were tested for use in solving the SLPCF. All data sets were adapted to determine the effects of the colour feasibility constraints in the performance of each algorithm.

Class	Data Set	Size (n, SKUs, m)	RSC	ASA	Data Set	Size (n, SKUs, m)	RSC	ASA
Large	A	(1262, 49, 49)	1237*	1237*	A′	(1262, 49, 59)	1008	997*
	B	(1264, 54, 54)	1226*	1226*	B′	(1264, 54, 64)	899*	937
	C	(1265, 51, 51)	1172	1166*	C′	(1265, 51, 61)	869*	906
	D	(1263, 56, 56)	1089*	1096	D′	(1263, 56, 66)	892*	914
	E	(1264, 51, 51)	1069*	1073	E′	(1264, 51, 61)	856*	894
	F	(1258, 53, 53)	1033	1026*	F′	(1258, 53, 63)	851*	902
	G	(1260, 56, 56)	984*	995	G′	(1260, 56, 66)	872*	880
	H	(1244, 54, 54)	998*	1015	H′	(1244, 54, 64)	954	948*
	I	(1264, 56, 56)	976	975*	I′	(1264, 56, 66)	854*	880
	J	(1258, 55, 55)	990*	991	J′	(1258, 55, 65)	756*	782
Medium	K	(943, 63, 63)	247*	281	K′	(943, 63, 73)	202*	244
	L	(846, 56, 56)	230*	236	L′	(846, 56, 66)	172*	174
	M	(728, 51, 51)	154*	157	M′	(728, 51, 61)	106*	108
	N	(396, 63, 63)	81*	138	N′	(396, 63, 73)	55*	123
	O	(733, 55, 55)	113*	122	O′	(733, 55, 65)	80*	92
	P	(242, 64, 64)	47*	79	P′	(242, 64, 74)	43*	76
	Q	(574, 48, 48)	78*	78*	Q′	(574, 48, 58)	44*	54
Small	R	(90, 48, 48)	7*	7*	R′	(90, 48, 58)	4*	4*
	S	(158, 55, 55)	14*	18	S′	(158, 55, 65)	9*	11
	T	(82, 51, 51)	8*	8*	T′	(82, 51, 61)	4*	7
	U	(80, 56, 56)	6*	7	U′	(80, 56, 66)	4*	5
	V	(89, 42, 42)	9	10	V′	(89, 42, 52)	5*	6

Table 6.3: *A comparison between the ant colony approach (ASA) and the random cluster sequencing approach with an appropriate clustering variation (RSC) in terms of the number of cycles traversed for the SLPCF. The best solution obtained is indicated by an asterisk. An upper bound may be seen as the number of orders n.*

From the results in Table 6.3 it is clear that there is no difference in performance between the RSC cluster sequencing approach and the ASA approach described by Algorithm 7 for large sized data sets where duplicated SKUs are not present. For medium and small sized data sets where duplicated SKUs are not present the RSC cluster sequencing approach is the preferred approach as it achieves the best solutions. This is also the case for all data sets where duplicated SKUs are present as the RSC cluster sequencing approach obtains the best solution in all cases except for data sets A′ and H′. This implies that the best approach to the SLPCF is the clustering approach.

6.2 SLP vs SLPCF

The SLPCF may be seen as a more restricted version of the SLP, however, there may not necessarily be an increase in the number of cycles traversed when solving the SLPCF. This may be attributed to the sub-optimal nature of heuristic and metaheuristic approaches. To determine the influence of the restriction the best solutions for both the SLP and SLPCF were compared in Table 6.4.

Class	Data Set	Size (n, SKUs, m)	SLP	SLPCF	Data Set	Size (n, SKUs, m)	SLP	SLPCF
	A	(1262, 49, 49)	1232	1237	A′	(1262, 49, 59)	925	1008
	B	(1264, 54, 54)	1226	1226	B′	(1264, 54, 64)	931	899
	C	(1265, 51, 51)	1161	1172	C′	(1265, 51, 61)	878	869
	D	(1263, 56, 56)	1072	1089	D′	(1263, 56, 66)	911	892
Large	E	(1264, 51, 51)	1069	1069	E′	(1264, 51, 61)	890	856
	F	(1258, 53, 53)	1005	1033	F′	(1258, 53, 63)	895	851
	G	(1260, 56, 56)	955	984	G′	(1260, 56, 66)	900	872
	H	(1244, 54, 54)	992	998	H′	(1244, 54, 64)	945	954
	I	(1264, 56, 56)	947	976	I′	(1264, 56, 66)	898	854
	J	(1258, 55, 55)	1025	990	J′	(1258, 55, 65)	778	756
	Total		**10684**	**10774**			**8951**	**8811**
	K	(943, 63, 63)	259	247	K′	(943, 63, 73)	269	202
	L	(846, 56, 56)	232	230	L′	(846, 56, 66)	194	172
	M	(728, 51, 51)	152	154	M′	(728, 51, 61)	114	106
Medium	N	(396, 63, 63)	90	81	N′	(396, 63, 73)	69	55
	O	(733, 55, 55)	125	113	O′	(733, 55, 65)	104	80
	P	(242, 64, 64)	45	47	P′	(242, 64, 74)	55	43
	Q	(574, 48, 48)	80	78	Q′	(574, 48, 58)	52	44
	Total		**983**	**950**			**857**	**702**
	R	(90, 48, 48)	7	7	R′	(90, 48, 58)	4	4
	S	(158, 55, 55)	14	14	S′	(158, 55, 65)	13	9
Small	T	(82, 51, 51)	8	8	T′	(82, 51, 61)	6	4
	U	(80, 56, 56)	6	6	U′	(80, 56, 66)	5	4
	V	(89, 42, 42)	9	9	V′	(89, 42, 52)	6	5
	Total		**44**	**44**			**34**	**26**

Table 6.4: *A comparison of the number of cycles traversed for the best SKU configuration for both the SKU location problem (SLP) and the SKU location problem with colour feasibility constraints (SLPCF). In both cases the best clustering variation was used for each class in conjunction with the RSC cluster sequencing approach with an appropriate clustering variation.*

Note that for many data sets the number of cycles traversed when solving the SLP exceeds that of the more restrictive SLPCF. This observation is true for most of the data sets where duplicated SKUs are present. This suggests that including colour feasibility constraints aid the clustering algorithms when solving the less constrained SLP.

The results presented show that the colour feasibility constraints would not significantly effect the number of cycles traversed in a picking line if the SLP is solved with the RSC cluster sequencing approach and therefore there is no trade off between reducing the picking risk and the number of cycles traversed. Furthermore, if duplicated SKUs are present it is suggested that the SLPCF be solved in place of the SLP when using the suggested RSC cluster sequencing approach for medium and small data sets.

6.3 Chapter Summary

DC policy established that SKUs which have the same colour and style and differ only in size may not be placed adjacent to each other in a picking line to prevent picking inaccuracies. This policy creates an additional constraint for the SLP and is handled in a revised problem formulation (SLPCF). Most of the approaches to the SLP were adapted for use in the SLPCF and it was found that clustering approaches perform the best, although different variations now show the best performance. Furthermore, it was found that the best clustering approach to the SLPCF outperformed the best approach for the SLP in many cases. It is therefore suggested that the SLPCF be solved in place of the SLP for all classes of data except the large sized data sets where duplicated SKUs are not present when using the suggested RSC cluster sequencing approach with an appropriate clustering variation.

CHAPTER 7

Results validation

Contents

All solutions to both the OSP and SLPCF subproblems rely on the assumptions that pickers may pass each other freely in the picking line and that the walking and picking speeds of the pickers are constant over all orders and locations. The total time required to complete a picking line could be reduced by only reducing the time spent walking. Both subproblems were therefore modelled with the objective of reducing the total number of cycles traversed and therefore the total walking time. In reality pickers might obstruct each other or may get congested in the picking line. For example, in most cases only a single picker may pick from a location at a time and pickers would have to queue when picking from the same location. To test the validity of the assumptions made to model the OSP and SLPCF a simulation model was built which took into account the interactions of pickers.

7.1 Simulation model

A continuous agent based simulation approach was used to simulate a picking line and was programmed using AnyLogic by XJ Technologies [36]. Agent based modelling (ABM) may be described as a simulation modelled by a collection of autonomous decision-making entities called agents [4]. Agents change their behaviour based on their current relationship with surrounding agents. This change in behaviour is usually governed by a set of simple rules. Although the behavioural rules may be quite simple for individual agents, more complex system patterns and behaviours emerge due to the interactions between agents.

One of the prime features of ABM is the use of heterogeneous agents to model self-organising systems. ABM is often used to model human systems and has been used to simulate consumer purchasing behaviour, the engagement of forces on a battlefield and customer flow management to name a few [24].

A typical structure of an agent-based model, according to Macal & North [24] consists of three elements:

1. A set of agents with attributes and behaviours.
2. A set of agent relationships and methods of interactions.
3. The agent's environment which also influences agent behaviour.

A picking line may be seen as an agent-based system with pickers as agents interacting with each other. Pickers also interact with the VRS and are confined to a physical space on the floor of a picking line. Both of these elements may be seen as the picker's environment. Each of the three elements of ABM proposed by Macal & North [24] are applied to the picking line problem considered here and are thus discussed further.

7.1.1 Agent attributes

After analysing video footage and the data of several picking lines it was concluded that a picker has three main tasks when achieving the goal of picking an order. The first is the physical movement of the picker to the location of the currently required SKU. The second is the physical picking of that SKU and the final activity is the handling of cartons, which include packing SKUs neatly in the carton, placing a carton on the conveyor belt and fetching a new carton. Based on these tasks each agent in the simulation model may find itself in one of three task states (TKS) namely:

1. Walking,
2. picking, or
3. carton handling.

Along with these possible states each agent has different attributes which determine the performance of that agent. Theses attributes include walking, picking and packing speeds and distributions. Furthermore, an agent will have a physical location in the picking line, a current order to be completed and the next SKU to be picked.

7.1.2 Agent relationships

When pickers come into close proximity of each other their behaviours change. Pickers will tend to follow slower pickers which are walking in the same direction instead of trying to pass them. When pickers do pass other pickers, which are in the process of picking items or packing cartons, they slow down as the walking area becomes narrower. In some cases pickers will stop altogether because there are to many other pickers in his/her immediate surrounding. Agents were therefore given the following behavioural states to model these changes of behaviour by the pickers:

1. Isolated,
2. following,
3. passing, and
4. congested.

An agent will be in an *isolated* state when there are no other agents within a predetermined distance in front of the agent. If an agent is in this behavioural state the agent assumes the default movement speed assigned to it at the start of the simulation. If an agent finds itself too close to at least two other agents the agents behaviour will change to the *congested* state and the agent will stop until the agents in front of it move on. If an agent is only close to a single agent two scenarios may arise: if the agent in front is picking or packing and the agent behind is moving towards a position passed the front agent, the agent behind will be in the *passing* state and it will reduce its speed as it passes the agent in front of it. If both agents are moving towards a location or are queued to pick at the same location the agent behind will be in a *following* state. When in a *following* state an agent assumes the slowest movement speed between itself and the agent in front of it. Figure 7.1 illustrates the logic diagram for determining an agent's behavioural state while Figure 7.2 illustrates possible scenarios where an agent is in one of the behavioural states.

7.1.3 Environment

The simulation environment emulates the physical layout of a picking line and the VRS system. The VRS keeps track of where each SKU is located and passes order information to the agents. The physical layout is based on measurements obtained from the DC and is designed to scale. It limits the movement paths of agents and influences the thresholds and behavioural states of an agent. For example, pickers slow down when passing other pickers due to the confined space between the conveyor belt and the locations.

7.1.4 Implementation

The simulation model was coded in both Anylogic developed by XJ Technologies [36] and Java [30]. A generic agent based simulation model was used to simulate the picker interactions and visualise the picking line. The interactions between agents were modelled using a flow diagram of different states. The sate of an agent is updated at regular time intervals. A screen shot of the implemented flow diagram as well as the visualisation of the picking line is given in Appendix B. The VRS was simulated with a custom plug in which simulated Algorithm 2. Each simulation required a set of picker speeds, SKU positions, a set of orders and a list of preferred starting positions for each order. Different solution approaches where tested by passing different SKU positions and preferred starting locations for the same set of SKUs and orders.

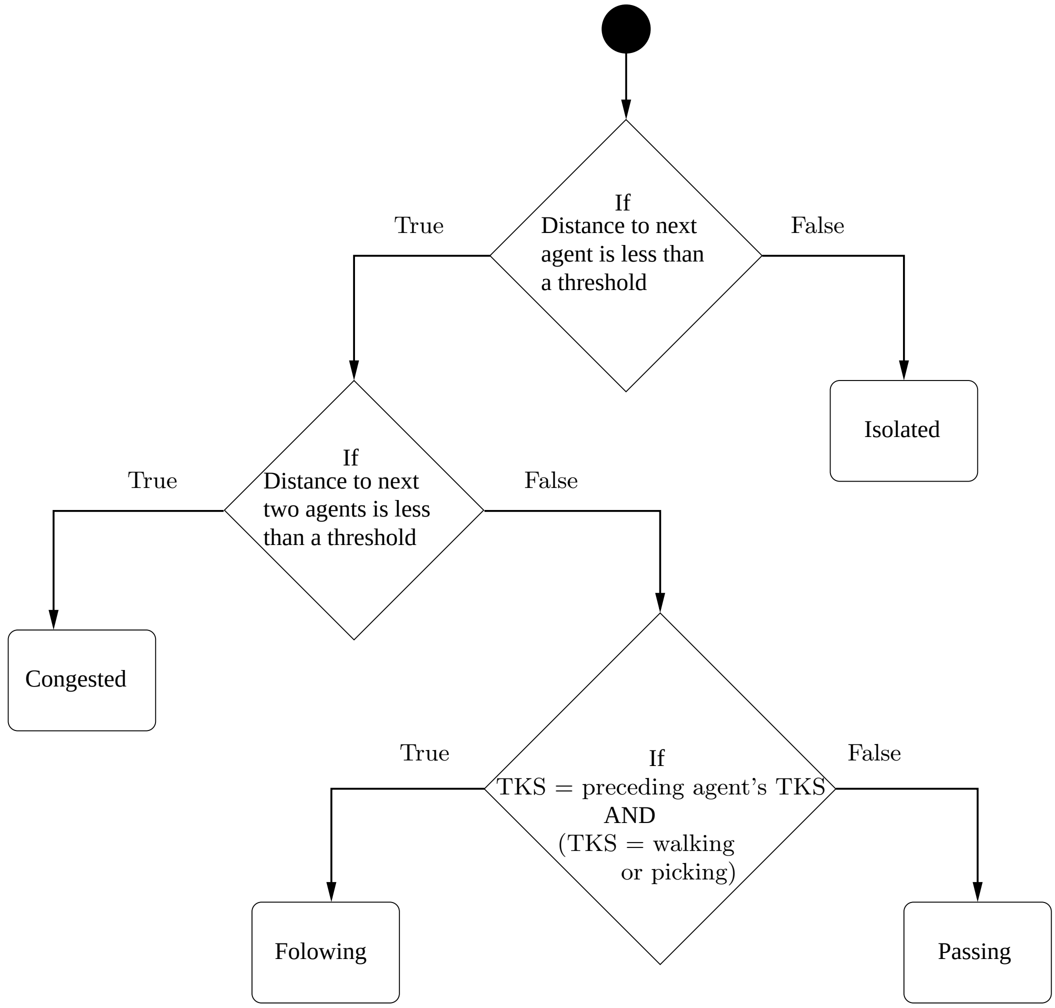


Figure 7.1: *A logic diagram to determine the behavioural state of an agent.*

7.2 Data capturing

An agent has three key attributes: a picking distribution, a packing distribution and a movement speed. The data used to test the OSP and SLPCF solution approaches did not contain detailed enough time stamp data to determine accurate distributions for pickers. The only time stamps available were the completion times of the specific picks which could not be used to determine the proportion of time spent on different activities. These time stamps could not be used to determine the time spent walking, picking and packing. The time stamp data also contains unknown congestion and passing scenarios as well as unknown delays such as bathroom breaks which influence the time taken to complete orders.

To determine these specific distributions for pickers video recordings of several additional picking lines were made and time stamps collected and analysed. Average picking and packing distributions were determined over all the pickers from these times using SAS [29]. Average free flow walking speeds were also calculated over all the pickers. It was also observed that after each pick there was a probability p for a picker to handle a carton. All the estimated data was not picker specific due to limited video data. These general estimations of distributions and speeds were used to estimate specific distributions for individual pickers by making the assumption that all pickers spend the same proportion of time picking, packing and walking. The general

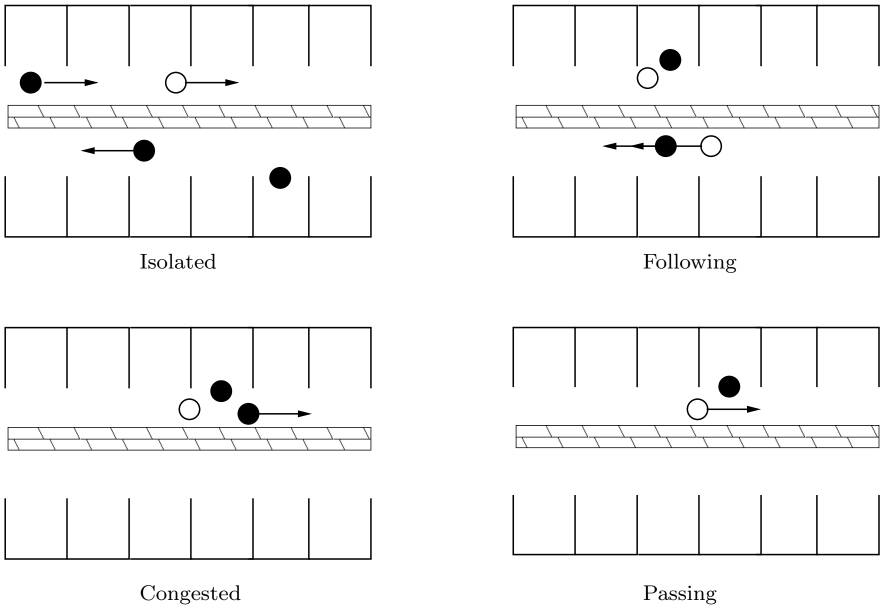


Figure 7.2: *Several scenarios where certain behavioural states can occur. The unfilled and filled dots represent agents. In each example the behavioural state of the unfilled agents are presented. The arrows indicates the direction in which the agents are moving. If no arrow is present the agent is packing or picking or waiting in a queue to pick.*

distributions were used to determine these proportions.

To estimate specific picking and packing distributions for individual pickers the total completion times for orders as well as the number of picks for each order were gathered from the historical time stamp data. The distributions for each picker were initially estimated by assuming that congestion did not take place and all the time taken to complete all the orders may be accounted for by walking, picking or packing. Once these initial distributions were established a test scenario was run based on the historical data with multiple pickers and the effects of congestion calculated. The time lost due to congestion was then used to adjust the walking, picking and packing distributions of each picker.

7.3 Verification and validation

Model verification is described by Sargent [28] as the determination of the correctness of the computational implementation of a model. The verification of the picking line simulation was done by using a set of simple static scenarios which modelled a representative set of model behaviours and characteristics. Two main modules where tested namely individual agent control and inter-agent relationships.

To test the individual agent control a single agent was sent through the model with a static list of orders. This tests whether agents moved in the correct directions and at the correct speeds. It also tested whether agents moved to the correct sequence of locations and completed all the orders. To test whether or not the VRS system is accurate several agents were used in a picking line to check whether all orders were picked, orders were assigned in the correct sequence (depending on the OSP solver) and all picks in each order were picked in the correct order.

The verification of the inter-agent relationships required the real time monitoring of several agents in the picking line. Initially two agents were placed in a picking line testing for the accurate calculation of the *following*, *isolated* and *passing* states. All four states where then tested by placing three and four agents in the picking line respectively. Two methods were used to test whether agents were in the correct states, firstly the animation was observed and secondly the individual states of each agent was monitored in conjunction with the positions of all the agents.

To validate the assumptions and test whether the simulation model behaves in a satisfactory manner compared to the reality several tests were performed. The simplest test to determine whether the behavioural states are valid is to run a simulation instance several times each time increasing the number of uniform agents in the system. The amount of time lost due to congestion should increase as the number of pickers increases. Table 7.1 illustrates the increase in total lost time due to congestion as the number of pickers increases. Although the total average wasted time for individual pickers decreases the percentage total time wasted increases as the number of pickers increases. The average wasted time per picker decreases because the total time spent in the picking line by the pickers decreases.

Number of Pickers	Average wasted time per picker	Average total wasted time	Percentage total time Wasted
1	0	0	0%
2	1914	3828	1.42%
3	1450	4350	1.61%
4	1133	4532	1.68%
5	1020	5100	1.89%
6	942	5652	2.09%
7	915	6405	2.36%
8	861	6888	2.53%
9	823	7407	2.71%
10	809	8090	2.96%

Table 7.1: *An illustration of the increase in time lost by pickers due to congestion as the number of pickers in a picking line is increased. In all cases a uniform set of pickers is used in conjunction with the same historical data set.*

The data used in the model, as discussed in § 7.2, was validated using historical data. Static picking scenarios were created where each agent was given the same fixed sequence of orders as in the historical data. In addition each agent was assigned a system entry time corresponding to the entry time of the picker in the historical data. Each test scenario was run 100 times and the individual completion times of the pickers were compared to the historical data. In some cases the data regarding certain orders were removed for specific pickers as the actual pick times were significantly large and were considered outliers. The large pick times may be accounted for by toilet breaks or other unforeseen interruptions.

Two statistical hypothesis tests were performed to determine the validity of the data. Firstly the total time required by an agent to complete its set of orders should correspond to the time taken historically by the corresponding picker. This was achieved by using the hypothesis

$$H_0 : \mu_i^s = \mu_i^h$$
$$H_A : \mu_i^s \neq \mu_i^h,$$

where μ_i^s is the average time required by the agent to complete all its orders and μ_i^h the historical time taken by the corresponding picker. The hypothesis testing was done using a 95% confidence level.

Furthermore, a test was performed to determine the spread of discrepancies between the time taken by an agent to pick a single order in comparison to the historical time by the actual picker. This was achieved by calculating the proportion of orders picked by the agent which required on average more time than the historical data. The expectation would be that half of the orders picked by the agent should require more time and half less. This was tested with the hypothesis

$$H_0 : p_i^o = 0.5$$
$$H_A : p_i^o \neq 0.5,$$

where p_i^o is the proportion of orders which on average took longer than the historical time.

Picker	Number of orders	μ_i^s	σ	μ_i^a	p_i
1	61	22848	511.10	23605	0.41
2	49	23018	388.40	23758	0.39
3	53	21347	680.22	21871	0.62
4	53	22955	554.48	23532	0.40
5	53	21210	501.07	21755	0.57
6	58	22591	433.13	23247	0.48
7	51	22423	595.32	23208	0.53
8	48	20843	713.3	21130	0.46

Table 7.2: *A table of the average and standard deviation of the simulated completion times (μ_i^s, σ), the actual completion time (μ_i^a) and the proportion of orders which were overestimated (p_i) by a set of pickers for the validation of the simulation model based on a single historical scenario. The scenario was run 100 times and all hypothesis testing did not reject the null hypothesis.*

A further measure to validate the data is comparisons of the individual walking speeds, picking and packing rates. It was noted that all the individual adjusted distributions for the different agents were within a 95% confidence interval of the average distributions obtained over all the pickers from the video data. This suggests that time lost by agents due to congestion, which was not explicitly calculated for pickers during data capturing, is valid. This result also suggests that the behavioural states model the congestion adequately.

7.4 Simulation scenarios

For both the OSP and SLPCF the objective function was in terms of cycles traversed by all the pickers. This objective function allowed for the direct comparison between the historical results obtained by Pep and the solutions to the respective subproblems. When considering the actual completion time for a picking line many stochastic elements are introduced as well as the issue of picker congestion. Within the time stamp data for each historical picking line there are a number of hidden elements which hinder the comparison of total historical completion time of a picking line to the completion times of simulated results. For example, in the historical results there are many instances where pickers join the picking line only for a fraction of the total completion time (in some cases a picker only picks 3 orders before leaving the picking line again). This results in insufficient data to model those specific pickers individually. In addition, unforeseen events may occur such as carton shortages (rare) or toilet breaks. Due to these hidden events to compare the current solution approaches used by Pep to the proposed methods both approaches are simulated under the same picker conditions.

To compare the results obtained in solving the OSP and SLPCF to the current approach used by Pep simulations were run using all the historical data sets. To allow for the comparison of results between different data sets a single set of picker distributions were used for all data sets. Eight pickers were used for each data set, as this was the number most frequently used in practice, and pickers enter the system at the same fixed rate until all pickers are in the system. All pickers continue to receive new orders until all orders are completed.

For each data set 4 scenarios were tested. The first scenario simulated the actual approach used by Pep where a fixed list of orders is given to the VRS and orders are sequentially passed to pickers as needed. The second scenario tests the effectiveness of the maximal cut approach to the OSP and determines if the objective of reducing only the number of cycles is viable. The simulated VRS is thus changed to make use of Algorithm 2. Two scenarios were tested for the SLPCF by considering different solution approaches to the corresponding OSP. The first scenario uses the exact solution to the maximal cut approach given in Formulation (4.20) to (4.24). The second scenario uses the solution obtained by using the HM variation of Algorithm 4. These two scenarios for the SLPCF will be referred to as the SLPCF Exact (SLPCFE) and SLPCF Metaheuristic (SLPCFM) and were chosen to determine the viability of using the HM variation of Algorithm 4 as an alternative approach which does not require IP solver software.

7.5 Results

All the scenarios were run for all the historical data sets. To obtain a reliable test each simulation was run 100 times with different random number sets. The first statistic to be tested is the total completion time of the picking lines. The average completion times as well as the standard deviations over the different simulations are shown in Table 7.3. Figure 7.3 illustrates the percentage difference between the different simulated solution approaches in comparison to Pep's simulated approach.

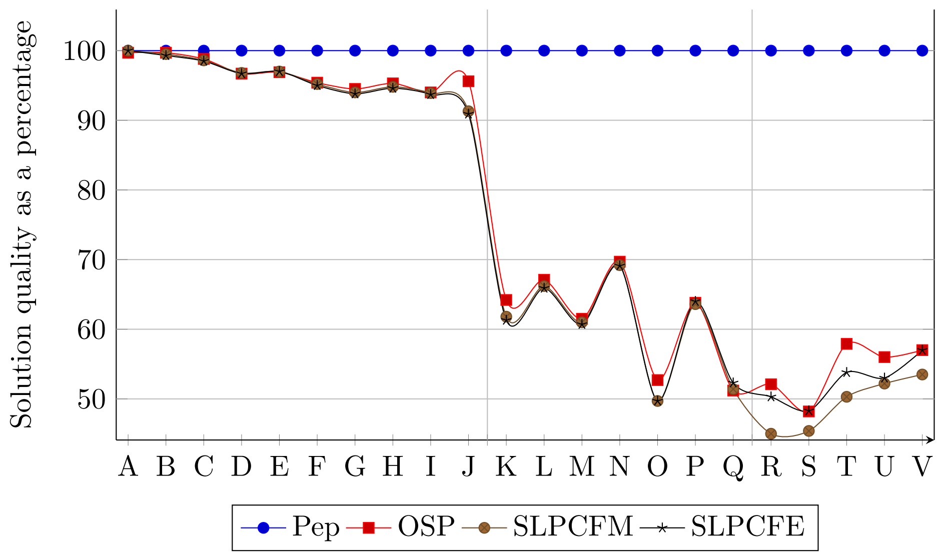


Figure 7.3: *A graphical illustration of the improvement as a percentage in total completion time when running simulations for the scenarios where Pep's current OSP philosophy is used (Pep), only the OSP is solved for historical SKU locations (OSP), the SLP is solved with the RSC cluster sequencing approach and the corresponding OSP solved with both the maximal cut approach (SLPCFE) and the HM variation of Algorithm 4 (SLPCFM). Peps approach represents the 100% benchmark.*

From the results presented in Table 7.3, which are summarised in Figure 7.3, it is clear that applying the maximal cut approach to solve the OSP yields shorter total completion times

		Size (n, SKUs, m)	Pep	OSP	SLPCFM	SLPCFE
Large	A	(1262, 49, 49)	(1038, 49)	(1036, 46)	(1038, 46)	(1039, 46)
	B	(1264, 54, 54)	(920, 47)	(917, 35)	(916, 39)	(913, 47)
	C	(1265, 51, 51)	(850, 45)	(840, 37)	(838, 41)	(838, 40)
	D	(1263, 56, 56)	(857, 40)	(829, 36)	(830, 40)	(829, 37)
	E	(1264, 51, 51)	(822, 42)	(796, 35)	(797, 37)	(797, 41)
	F	(1258, 53, 53)	(812, 45)	(775, 36)	(773, 39)	(772, 37)
	G	(1260, 56, 56)	(814, 44)	(770, 37)	(766, 38)	(764, 40)
	H	(1244, 54, 54)	(884, 41)	(843, 37)	(838, 36)	(836, 37)
	I	(1264, 56, 56)	(777, 44)	(730, 42)	(730, 32)	(728, 37)
	J	(1258, 55, 55)	(675, 36)	(645, 30)	(616, 31)	(613, 35)
Medium	K	(943, 63, 63)	(230, 21)	(147, 15)	(142, 14)	(141, 17)
	L	(846, 56, 56)	(182, 17)	(122, 13)	(121, 14)	(120, 14)
	M	(728, 51, 51)	(131, 30)	(80, 14)	(80, 13)	(79, 28)
	N	(396, 63, 63)	(100, 18)	(70, 12)	(69, 17)	(69, 18)
	O	(733, 55, 55)	(123, 14)	(65, 10)	(61, 11)	(61, 10)
	P	(242, 64, 64)	(59, 17)	(38, 16)	(38, 14)	(38, 15)
	Q	(574, 48, 48)	(80, 18)	(41, 8)	(41, 8)	(42, 9)
Small	R	(90, 48, 48)	(12, 6)	(6, 5)	(5, 3)	(6, 5)
	S	(158, 55, 55)	(24, 7)	(11, 6)	(10, 5)	(11, 6)
	T	(82, 51, 51)	(12, 16)	(7, 10)	(6, 6)	(6, 8)
	U	(80, 56, 56)	(13, 21)	(7, 13)	(7, 16)	(7, 14)
	V	(89, 42, 42)	(11, 6)	(6, 5)	(6, 6)	(6, 5)

Table 7.3: *The average and standard deviation* (μ, σ) *of the total completion time in minutes for scenarios where Pep's current OSP philosophy is used (Pep), only the OSP is solved for historical SKU locations (OSP), the SLP is solved with the RSC cluster sequencing approach and the corresponding OSP solved with both the maximal cut approach (SLPCFE) and the HM variation of Algorithm 4 (SLPCFM) for historical data sets. A hundred instances of each simulation were run and the results are given in minutes rounded to the nearest minute.*

in comparison to the approach used by Pep. This suggests that the objective of reducing the number of cycles traversed in a picking line is a valid objective which reduces the total completion time. The results also suggest that there is no significant improvement in the completion time of a picking line if the SLPCF is solved. Only data set J shows a significant improvement when solving the SLPCF instead of just the OSP. This suggests that the original location assignments used by Pep yielded poor solutions to the SLP. The results for the SLPCFM and SLPCFE scenarios are not significantly different suggesting that the use of the HM variation of Algorithm 4 as a solution approach is viable.

Not only does solving the OSP with the maximal cut approach reduce the overall completion time of the picking line it also reduces the variances in completion time. This suggests that a better estimation of how long a picking line should take to be completed may be made when optimising the OSP. A better estimation of expected completion times assists in a more accurate calculation of picker key performance indicators (KPIs). It is also suggested that the OSP is the main contributor to the reduced completion times as well as the reduced variances.

One of the main influences on the decision by management to use 8 pickers in picking line is the time lost due to picker congestions. This lost time occurs whenever a picker is in a queue, following a slower picker, stuck at a point of congestion or passing another picker. The results for the total lost time are presented in Table 7.4.

		Size (n, SKUs, m)	Pep	OSP	SLPCFM	SLPCFE
Large	A	(1262, 49, 49)	(32, 31)	(32, 31)	(32, 31)	(33, 31)
	B	(1264, 54, 54)	(30, 29)	(30, 30)	(30, 28)	(29, 28)
	C	(1265, 51, 51)	(30, 29)	(29, 32)	(27, 26)	(28, 25)
	D	(1263, 56, 56)	(28, 27)	(26, 28)	(24, 24)	(24, 21)
	E	(1264, 51, 51)	(26, 24)	(25, 24)	(24, 24)	(24, 25)
	F	(1258, 53, 53)	(27, 24)	(24, 24)	(24, 24)	(24, 23)
	G	(1260, 56, 56)	(27, 25)	(23, 21)	(21, 19)	(20, 21)
	H	(1244, 54, 54)	(28, 24)	(24, 24)	(23, 23)	(23, 23)
	I	(1264, 56, 56)	(25, 25)	(22, 21)	(22, 22)	(22, 21)
	J	(1258, 55, 55)	(27, 28)	(25, 24)	(22, 22)	(22, 21)
Medium	K	(943, 63, 63)	(7, 7)	(5, 5)	(4, 5)	(4, 5)
	L	(846, 56, 56)	(6, 5)	(4, 4)	(5, 5)	(5, 5)
	M	(728, 51, 51)	(5, 5)	(3, 3)	(3, 4)	(3, 3)
	N	(396, 63, 63)	(4, 4)	(2, 3)	(2, 2)	(2, 2)
	O	(733, 55, 55)	(4, 4)	(3, 3)	(3, 2)	(3, 3)
	P	(242, 64, 64)	(2, 2)	(2, 1)	(2, 2)	(1, 2)
	Q	(574, 48, 48)	(4, 4)	(2, 2)	(2, 2)	(2, 2)
Small	R	(90, 48, 48)	(1, 1)	(1, 1)	(1, 1)	(1, 1)
	S	(158, 55, 55)	(1, 1)	(1, 1)	(1, 1)	(1, 1)
	T	(82, 51, 51)	(1, 1)	(1, 1)	(1, 1)	(1, 1)
	U	(80, 56, 56)	(1, 1)	(1, 1)	(1, 1)	(1, 1)
	V	(89, 42, 42)	(1, 1)	(1, 1)	(1, 1)	(1, 1)

Table 7.4: *The average total lost time (expressed as average, standard deviation) of a picker due to obstructions by other pickers for the scenarios where Pep's current OSP philosophy is used (Pep), only the OSP is solved for historical SKU locations (OSP), the SLP is solved with the RSC cluster sequencing approach and the corresponding OSP solved with both the maximal cut approach (SLPCFE) and the HM variation of Algorithm 4 (SLPCFM) using historical data sets. A hundred instances of each simulation were run and the results are given in minutes rounded to the nearest minute.*

The results in Table 7.4 suggest that implementing the algorithms to solve the OSP or SLPCF does not increase the average time lost due to picker obstructions. It may also be seen that a correlation exists between the size of a picking line and the number of minutes lost due to picker obstructions. It should be noted that the standard deviation for the time lost due to picker obstructions is significantly large ($\sigma \approx \mu$) suggesting that this lost time may vary significantly between pickers as well as different simulations. The percentage time lost is, however, significantly small (roughly 3%) and it is therefore deduced that the effects of picker obstructions are insignificant when determining a picker's performance.

One of the considerations when designing an order pick system is that of work balance between pickers. Currently the KPI for a picker is the number of picks per time unit. This KPI does not take into account the sequence or type of orders which are assigned to a picker. A key statistic to determine whether work is being assigned in a balanced manner is the number of picks per cycle. The results in Table 7.4 suggest that the effect of picker obstructions are insignificant when determining picker performance and therefore if the number of picks per cycle for each picker are equal the KPI would be effective at indicating the performance of individual pickers. The average picks per cycle as well as the standard deviation between pickers for each simulation is given in Table 7.5.

		Size (n, SKUs, m)	Pep	OSP	SLPCFM	SLPCFE
Large	A	(1262, 49, 49)	(20.04, 0.05)	(20.28, 0.26)	(20.3, 0.08)	(20.35, 0.25)
	B	(1264, 54, 54)	(16.58, 0.09)	(16.83, 0.14)	(16.87, 0.09)	(16.9, 0.17)
	C	(1265, 51, 51)	(15.43, 0.04)	(16.24, 0.3)	(16.29, 0.11)	(16.33, 0.12)
	D	(1263, 56, 56)	(15.46, 0.06)	(17.36, 0.07)	(17.07, 0.09)	(17.2, 0.11)
	E	(1264, 51, 51)	(15.13, 0.05)	(17.02, 0.08)	(17.01, 0.11)	(17.08, 0.11)
	F	(1258, 53, 53)	(14.86, 0.03)	(17.55, 0.1)	(17.51, 0.13)	(17.62, 0.08)
	G	(1260, 56, 56)	(14.99, 0.08)	(18.36, 0.13)	(18.41, 0.1)	(18.56, 0.05)
	H	(1244, 54, 54)	(16.66, 0.05)	(19.94, 0.1)	(20.34, 0.11)	(20.46, 0.1)
	I	(1264, 56, 56)	(14.02, 0.07)	(17.32, 0.1)	(17.36, 0.08)	(17.46, 0.15)
	J	(1258, 55, 55)	(11.38, 0.03)	(12.94, 0.06)	(14.9, 0.11)	(15.09, 0.06)
Medium	K	(943, 63, 63)	(4.19, 0.03)	(9.94, 0.17)	(10.84, 0.19)	(11.04, 0.14)
	L	(846, 56, 56)	(4.21, 0.04)	(9.4, 0.2)	(9.81, 0.09)	(9.8, 0.34)
	M	(728, 51, 51)	(3.58, 0.15)	(9.75, 0.39)	(10.12, 0.4)	(10.12, 0.54)
	N	(396, 63, 63)	(6.34, 0.08)	(14.9, 0.24)	(15.34, 0.3)	(15.34, 0.33)
	O	(733, 55, 55)	(2.72, 0.02)	(8.88, 0.12)	(10.16, 0.36)	(10.31, 0.21)
	P	(242, 64, 64)	(5.39, 0.15)	(15.53, 0.54)	(15.84, 1.2)	(15.01, 0.61)
	Q	(574, 48, 48)	(2.46, 0.03)	(8.69, 0.1)	(8.62, 0.27)	(8.23, 0.18)
Small	R	(90, 48, 48)	(1.92, 0.17)	(6.51, 0.75)	(15.77, 5.08)	(8.32, 1.04)
	S	(158, 55, 55)	(2.45, 0.2)	(10.57, 0.73)	(12.18, 1.33)	(10.31, 1.38)
	T	(82, 51, 51)	(2.56, 0.17)	(8.36, 1.32)	(34.4, 10.59)	(12.31, 3.47)
	U	(80, 56, 56)	(3.01, 0.2)	(10.79, 1.89)	(19.39, 3.97)	(15.7, 2.56)
	V	(89, 42, 42)	(2.05, 0.09)	(7.01, 1.56)	(10.85, 2.79)	(6.82, 1.12)

Table 7.5: *The average picks per cycle and standard deviation (μ, σ) of pickers for the scenarios where Pep's current OSP philosophy is used (Pep), only the OSP is solved for historical SKU locations (OSP), the SLP is solved with the RSC cluster sequencing approach and the corresponding OSP solved with both the maximal cut approach (SLPCFE) and the HM variation of Algorithm 4 (SLPCFM) using historical data sets. A hundred instances of each simulation were run and the results are given in picks per cycle.*

From the results in Table 7.5 it is suggested that there is a trade off between the efficiency of a picking line and the effective balancing of work. The trade off, however, does not appear to be significant as the standard deviations relative to the averages are low.

It should be noted that the picks per cycle for the SLPCFM is greater than that of the SLPCFE for small data sets. This may be explained by the size of the data sets and the rigidity of the order sequence associated with the SLPCFE solution in conjunction with Algorithm 2. The stochastic nature of multiple pickers appears to have a greater effect when the data sizes are small. Although the picks per cycle differ significantly the total completion time, shown in Table 7.3, appears to have no significant difference.

To determine the significance of the correlation between the number of cycles and the completion time the percentage improvements in the number of cycles is compared to that of the completion time. Finding a good correlation between number of cycles and completion time may assist in the comparison of different picking lines.

		Size (n, SKUs, m)	OSP	SLPCFM	SLPCFE
Large	A	(1262, 49, 49)	(1.3%, 0.27%)	(1.3%, 0.01%)	(1.4%, -0.03%)
	B	(1264, 54, 54)	(1.4%, 0.35%)	(1.8%, 0.5%)	(1.9%, 0.74%)
	C	(1265, 51, 51)	(5.1%, 1.24%)	(5.2%, 1.4%)	(5.4%, 1.47%)
	D	(1263, 56, 56)	(11%, 3.34%)	(9.5%, 3.19%)	(10.1%, 3.3%)
	E	(1264, 51, 51)	(11.1%, 3.06%)	(11%, 2.95%)	(11.4%, 3.04%)
	F	(1258, 53, 53)	(15.3%, 4.6%)	(15.2%, 4.79%)	(15.7%, 4.98%)
	G	(1260, 56, 56)	(18.3%, 5.47%)	(18.6%, 5.97%)	(19.2%, 6.16%)
	H	(1244, 54, 54)	(16.5%, 4.7%)	(18.1%, 5.2%)	(18.6%, 5.4%)
	I	(1264, 56, 56)	(19%, 6.04%)	(19.2%, 6.1%)	(19.7%, 6.34%)
	J	(1258, 55, 55)	(12.1%, 4.39%)	(23.7%, 8.72%)	(24.6%, 9.14%)
Medium	K	(943, 63, 63)	(57.9%, 35.8%)	(61.4%, 38.23%)	(62.1%, 38.68%)
	L	(846, 56, 56)	(55.3%, 32.9%)	(57.1%, 33.78%)	(57.2%, 34.08%)
	M	(728, 51, 51)	(63.2%, 38.45%)	(64.6%, 38.99%)	(64.6%, 39.33%)
	N	(396, 63, 63)	(57.4%, 30.28%)	(58.6%, 30.77%)	(58.6%, 30.92%)
	O	(733, 55, 55)	(69.4%, 47.26%)	(73.3%, 50.28%)	(73.7%, 50.29%)
	P	(242, 64, 64)	(65.2%, 36.15%)	(65.9%, 36.43%)	(64.1%, 35.96%)
	Q	(574, 48, 48)	(71.7%, 48.76%)	(71.5%, 48.62%)	(70.1%, 47.71%)
Small	R	(90, 48, 48)	(70.3%, 47.9%)	(87.7%, 54.99%)	(76.7%, 49.72%)
	S	(158, 55, 55)	(76.7%, 51.8%)	(79.9%, 54.57%)	(76.2%, 51.72%)
	T	(82, 51, 51)	(69.1%, 42.11%)	(92.2%, 49.66%)	(78.5%, 46.18%)
	U	(80, 56, 56)	(72%, 43.96%)	(84.5%, 47.75%)	(80.9%, 47.02%)
	V	(89, 42, 42)	(69.6%, 43.02%)	(81.2%, 46.54%)	(69.9%, 42.98%)

Table 7.6: *The average percentage improvement of cycles and average percentage improvement of completion time relative to the approach used by Pep (Pep) for the scenarios where only the OSP is solved for historical SKU locations (OSP), the SLP is solved with the RSC cluster sequencing approach and the corresponding OSP solved with both the maximal cut approach (SLPCFE) and the HM variation of Algorithm 4 (SLPCFM) using historical data sets. Each simulation was run 100 times and the results are given in percentages.*

The results presented in Table 7.6 suggest that for large picking lines the time savings are approximately $\frac{1}{3}$ of the savings in terms of number of cycles traversed. For medium sized picking lines this proportion is approximately $\frac{1}{2}$ and for the small picking lines $\frac{2}{3}$. These proportions of savings may be seen as the proportion of time spent walking in the picking lines. Proportionally less time is spent walking in the larger picking lines as there are more orders and more picks in each order. The percentage improvement in terms of cycles is summarised in Figure 7.4.

7.6 Chapter Summary

To test the validity of only minimising the number of cycles traversed in both the OSP and SLPCF approaches a simulation model was built. The simulation may be seen as a continuous agent based simulation model where the pickers are modelled as agents with specific attributes

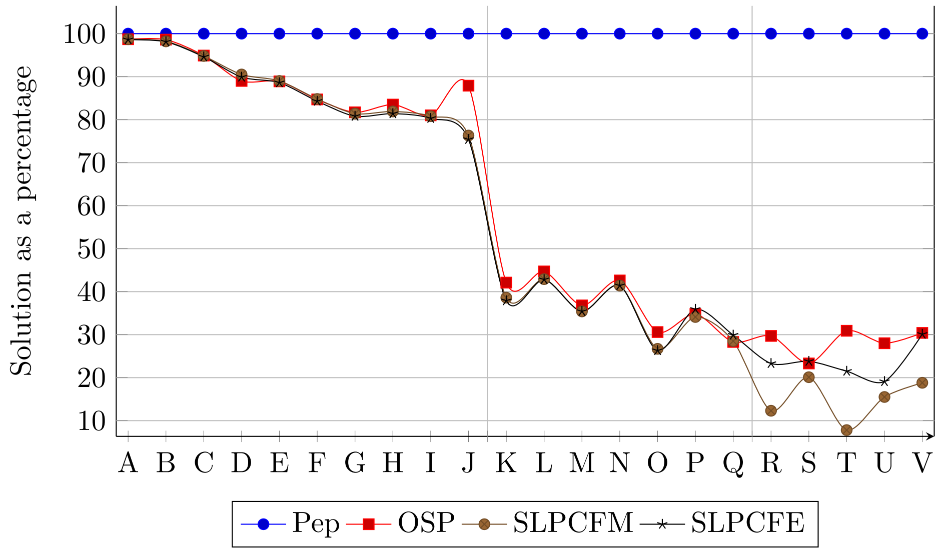


Figure 7.4: *A graphical illustration of the improvement as a percentage in cycles traversed relative to the approach used by Pep (Pep) for the scenarios where only the OSP is solved for historical SKU locations (OSP), the SLP is solved with the RSC cluster sequencing approach and the corresponding OSP solved with both the maximal cut approach (SLPCFM) and the HM variation of Algorithm 4 (SLPCFM) using historical data sets. Pep's approach represents the 100% benchmark.*

and behavioural patterns which are affected by other pickers and the environment. Data capturing was done making use of video footage to estimate the picking and packing distributions as well as the walking speeds of pickers. These initial estimates where adjusted for a set of pickers using historical data.

Four different scenarios were tested for each historical data set. These scenarios were chosen to determine differences in completion times between Pep's approach, the OSP solutions and the SLPCF solutions. It was found that the objective of reducing the number of cycles traversed in a picking line is valid. In addition time lost due to the obstructions of other pickers was seen to remain constant regardless of which solution approach was used. It was also found that a trade off existed between the efficiency of a picking line and the balancing of work between pickers. This trade off was, however, not seen as significant relative to increases in the efficiency when solving the OSP.

CHAPTER 8

Conclusions

Contents

A brief summary of the work done in this thesis is provided in this Chapter. In addition final conclusions and remarks are given, as well as a discussion of potential future studies and the use of the results in these future studies.

8.1 Thesis summary

A general overview of DCs, a literature review of order picking and the thesis scope and objectives is given in Chapter 1. The purpose of Chapter 2 was to explain the operations of the DC as well as the order picking process and management system. Detailed descriptions of the physical layout of the DC, the order picking processes as well as the management system were provided.

In Chapter 3 both physical and managerial constraints were identified and appropriate assumptions made. The order pick operation was identified as having 3 hierarchical levels of decision making. These three levels of decision making lead to the definition of three subproblems (OSP, SLP, PLAP). The definition of each and the relationship between these subproblems were briefly discussed in §3.2.

In Chapter 4 the OSP was modelled and solution approaches developed. Appropriate assumptions were made which transformed a time based objective to one of distance. An assumption was also made which allowed for the problem to be modelled as if only a single picker were in the picking line. The OSP was modelled as an equality generalized travelling salesman problem and an exact formulation was presented. The computational effort required to solve the exact formulation is too large for real life data sets. This led to the investigation of heuristic and metaheuristic methods.

To determine the effectiveness of these algorithms a good lower bound was achieved by using the concept of a maximal cut. The solution to the lower bound or maximal cut approach could be transformed to a feasible solution within 1 cycle of the lower bound to the OSP. This maximal cut approach is discussed in §4.3. The computation time for the maximal cut approach was still too long for the use in the SLP. Thus four greedy algorithms, a random local search and hybrid thereof were developed and discussed in §4.5. The hybrid of the random local search showed the best performance in terms of solution quality but the greedy heuristics showed significantly faster computation times with reasonable solution qualities.

In Chapter 5 the modelling and solving of the SLP is discussed. In §5.2 two known heuristics for product locations on bidirectional carousels, namely the Organ Pipe approach (OP) and a Greedy approach (GP), were tested and adapted for cases where duplicated SKUs were present. In addition an ant colony approach (ASA), presented in §5.3, was developed making use of principles applied to TSPs and variants thereof.

In an attempt to reduce the problem size and the size of the solution space visited by local search and solution generation techniques, clustering algorithms were developed to cluster different SKUs together. Four clustering variations, discussed in §5.4, were tested and the best one for different data classes used. Once SKUs were placed into clusters three cluster sequencing approaches were tested to arrange these different clusters namely a random search (RSC), ant colony variation (ACC) and a Tabu search (TC).

The addition of colour feasibility constraints in the SLP definition (SLPCF) is discussed in Chapter 6. The formulation to the SLP was modified and most of the approaches to the SLP were adapted for use in the SLPCF. It was found that in many cases the solutions of, the more restrictive, SLPCF were better than those of the SLP when solving with the RSC algorithm. It was therefore suggested that the SLPCF be solved by the RSC when solving the SLP for all data classes except for large data sets where duplicated SKUs are not present.

The purpose of Chapter 7 was to test the validity of using the minimisation of the number of cycles traversed as an objective function and compare the solution approaches to the current methods used by Pep. An agent based simulation model was built and is discussed in §7.1. Four different scenarios were tested for each historical data set. These scenarios were chosen to determine differences in completion times between Pep's approach, the OSP solutions and the SLPCF solutions. It was found that the objective of reducing the number of cycles traversed in a picking line is valid and that the solutions obtained by solving the OSP and SLPCF were significantly better than the approaches used by Pep.

8.2 Recommendations

The maximal cut approach was developed to solve the OSP and Algorithm 2 introduced to dynamically allocate orders to multiple pickers in real time. Some solution quality is lost when moving from a single picker to multiple pickers, however, it was shown in §4.4 that this loss is negligible. Currently Pep uses a random sequencing approach for the OSP and it was shown that the maximal cut approach significantly outperforms Pep's approach. It it therefore recommended that Pep use the maximal cut approach in conjunction with Algorithm 2 within the VRS to allocate orders to pickers.

In §7.5 the SLPCF solutions were compared to the solutions obtained by solving the OSP using the historical SKU locations used by Pep. The results suggest that the greatest improvement to the picking times occurs when solving the OSP and that the historical SKU locations yield good solutions. It is therefore suggested that Pep may choose whether to use the RSC approach to the SLPCF or continue to use the current system.

8.3 Future work

The scope of this thesis was to describe the order picking operation at Pep, define the global problem and subproblems and to solve the first two subproblems. A natural continuation of this study would be to investigate possible solution methods for the picking line allocation problem (PLAP) discussed in §3.2. The PLAP relates to the SLP in the same way as the SLP does to the OSP. The PLAP thus inherently relies on, and to an even greater capacity, the solution approaches to the OSP. It may therefore be useful to investigate the possibility of determining other measures of a picking lines efficiency than continuously recalculation the OSP.

Another scenario which has been considered throughout the thesis is the inclusion of duplicated SKUs. Algorithms were developed which could handle these duplicated SKUs to be used in feasibility studies to determine if duplicating SKUs on a picking line could improve overall efficiency. Both the solution approaches to the OSP and SLP may assist in this study as well as the simulation model discussed in Chapter 7.

Many of the constraints imposed on the problems were due to the short term rigidity of the DC and picking line layout. Natural questions arise when considering long term planning such as:

1. What should the ratio of picking lines to DC size be?
2. How big should the picking lines be?
3. What is the optimum picking line mix?

These questions will require a more long term effort and foresight as markets are ever-changing. The answers, however, could prove useful in future DC development and layout of Pep.

One of the most influential management decisions on the order picking system is the FIFO methodology when determining which SKUs should be picked next. A useful study would be to develop a system of determining preferred time windows when SKUs should be picked. Although a time window approach would grant more freedom to solution approaches to the PLAP the determination of these time windows would have to consider a wider range of variables including, the seasonality of stock, DC space and the risk of lost sales at retail outlets, to name a few.

The DC forms only part of the broader supply chain operations of Pep. Therefore it would be useful to investigate the downstream effects of DC decisions on the supply chain. The effects include transportation to, and holding cost at retail outlets. The results of such a study could impact the planning in the DC and central office.

8.4 Thesis objectives

In §1.4 the following objectives were identified:

Objective I

a To describe the layout and operations of the DC so that the problem may be viewed in the broader DC context;

b To describe the order picking system in detail so that the characteristics of the problem may be understood;

Objective II

a Identify long term and short term problem constraints and make suitable assumptions so that a detailed problem may be identified and modelled;

b Identify all levels of decision making in the order sequencing operation;

Objective III

a Make suitable assumptions to model and solve the order sequencing subproblem;

b Make suitable assumptions to model and solve the SKU location subproblem;

Objective IV

a Develop a simulation model to test solution approaches of both the order sequencing and SKU location subproblems;

b Compare results to actual approaches used by Pep;

Objective V

a Discuss potential directions of future studies;

Objective I was reached in Chapter 1 where a detailed discussion was made regarding DC s with special focus on the DC owned by Pep. A detailed discussion regarding the order picking system used by Pep is also given. In Chapter 3 the major managerial, system and physical constraints affecting the order picking system were identified. Appropriate assumptions were made after consultation with management at Pep. This was done in fulfilment of objective II. The order sequencing problem (OSP) is discussed and solved in Chapter 4. Specific assumptions regarding the OSP were made and several solution procedures developed. The solution procedures were shown to be robust and practical. In Chapters 5 and 6 two cases of the SLP were considered. The first case allowed for any two SKUs to be placed adjacent to each other (SLP) while the second case (SLPCF) incorporated the colour feasibility constraints which are constraints imposed by management. Objective III was therefore reached. In Chapter 7 an agent based simulation model is introduced to validate the solutions to the OSP and SLP in fulfilment of objective IV. It was shown that the original assumptions made are plausible and that the solution procedures recommended may be used in practice. Finally objective V is fulfilled in Chapter 8.

8.5 Contributions

An unsolved problem from industry was investigated in this thesis. The OSP had many similarities to a unidirectional carousel system which is a rare problem in literature. A novel tight lower bound to the problem was developed and it was proven that a feasible solution to the OSP maybe found within 1 cycle of this lower bound by using a maximal cut approach. Furthermore, it was shown that only starting locations for orders are required to solve the OSP and that the subtour generation constraints are non-binding. This reduced problem complexity considerably and aided in the transition to multiple pickers and was one of the major breakthroughs as it made the problem tractable.

The OSP was defined as the sequencing of a set of orders for a single picker. The maximal cut approach was shown to be dynamic and robust for use in the case of multiple pickers. A suitable algorithm was developed to assign orders in real-time to pickers on a picking line. The solution was shown to have insignificant decreases in solution quality when moving from a sequence for a single picker to multiple pickers.

Although the maximal cut approach yields solutions close to optimal the SLP requires faster solution procedures for the OSP. Several novel fast heuristics were developed based on a greedy sequencing approach. Four novel distance measures were developed and tested for use in the greedy sequencing framework.

The results of this study showed that the maximal cut approach to order sequencing is significantly better than the approaches used by Pep. Due to the results and the robust nature of the maximal cut approach Pep is in the process of implementing the proposed order sequencing approach.

Bibliography

[1] AMAZON, 2010, *Amazon.com*, [Online], [Cited February 22nd, 2011], Available from http://www.amazon.com/.

[2] BARTHOLDI J & HACKMAN S, 2007, *Warehouse and distribution science*, [Online], [Cited May 1st, 2009], Available from http://warehouse-science.com/.

[3] BARTHOLDI JJ & PLATZMAN LK, 1986, *Retrieval strategies for a carousel conveyer*, IIE Transactions, **18(10)**, pp. 166–173.

[4] BONABEAU E, 2002, *Agent-based modeling: Methods and techniques for simulating human systems*, PNAS, **99(3)**, pp. 7280–7287.

[5] BOWERSOX DJ, CLOSS DJ & COOPER MB, 2007, *Supply Chain Logistics Management*, McGrawHill, New York (NY).

[6] CAMAZINE S, DENEUBOURG JL, FRANKS NR, SNEYD J, THERAULAZ G & BONABEAU E, 2001, *Self-Organization in Biological Systems*, Princeton University Press, Princeton.

[7] CHENG C & MAO C, 2007, *A modified ant colony system for solving the travelling salesman problem with time windows*, Mathematical and Computer Modelling, **46(9-10)**, pp. 1225–1235.

[8] DAIFUKU, 2011, [Cited April 18th, 2011], Available from http://www.daifukuamerica.com.

[9] DE KOSTER R, LE-DUC T & ROODBERGEN KJ, 2007, *Design and control of warehouse order picking: A literature review*, European Journal of Operational Research, **182**, pp. 481–501.

[10] FERRET, 2011, [Cited April 18th, 2011], Available from http://www.ferret.com.au/c/Storage-Ideas.

[11] FISCHETTI M, SALAZAR-GONZÁLEZ JJ & TOTH P, 2007, *The generalized traveling salesman and orienteering problems*, pp. 609–662 in GUTIN G & PUNNEN AP (EDS), *The traveling salesman problem and its variations*, Springer, New York (NY).

[12] FRAZELLE EH, 2002, *World Class Warehousing and Material Handling*, McGrawHill, New York (NY).

[13] GLOVER F, 1986, *Future paths for integer programming and links to artificial intelligence*, Computers and Operations Research, **13(5)**, pp. 533–549.

[14] Grant DB, Lambert DM, Stock JR & Ellram LM, 2006, *Fundamentals of Logistics*, McGrawHill, New York (NY).

[15] Handling B, 2011, [Cited April 18th, 2011], Available from http://www.handling.barloworld.com/equipment/new-fork-lifts/default.asp.

[16] Hassini E, 2009, *One-dimensional carousel storage problems: Applications, review and generalizations.*, INFOR, **47(2)**, pp. 81 – 92.

[17] Hsu CM, Chen KY & Chen MC, 2005, *Batching orders in warehouses by minimizing travel distance with genetic algorithms*, Computers in Industry, **56(2)**, pp. 169–178.

[18] Integrators A, 2011, [Cited April 18th, 2011], Available from http://www.autointegrators.com/.

[19] Jewkes E, Lee C & Vickson R, 2004, *Product location, allocation and server home base location for an order picking line with multiple servers*, Computers & Operations Research, **31(4)**, pp. 623–636.

[20] Kaufman L & Rousseeuw PJ, 2005, *Finding Groups in Data An Introduction to Cluster Analysis*, John Wiley & Sons, Inc., Hoboken New Jersey.

[21] Lindo Systems, 2010, *Lingo 11*, [Online], [Cited May 1st, 2011], Available from http://www.lindo.com/.

[22] Litvak N & Adan I, 2002, *On a class of order pick strategies in paternosters*, Operations Research Letters, **30**, pp. 377 – 386.

[23] Logistex, 2011, [Cited April 18th, 2011], Available from http://www.logisticsit.com/absolutenm/templates/article-wms.aspx?articleid=4965.

[24] Macal CM & North MJ, 2010, *Tutorial on agent-based modelling and simulation*, Journal of Simulation, **4**, pp. 151–162.

[25] Pan CH & Liu SY, 1995, *A comparative study of order batching algorithms*, International Journal of Managment Science, **23(6)**, pp. 691–700.

[26] Pep stores, 2010, *Pep*, [Online], [Cited December 4th, 2010], Available from http://www.pepstores.com.

[27] Punnen AP, 2007, *The traveling salesman problem: Applications, forumlations and variations*, pp. 1–28 in Gutin G & Punnen AP (Eds), *The traveling salesman problem and its variations*, Springer, New York (NY).

[28] Sargent RG, 1998, *Verification and validation of simulation models*, Proceedings of the 30th conference on Winter simulation IEEE Computer Society, Los Alamitos CA, pp. 121–130.

[29] SAS, SAS 9.1, [Online], [Cited November 21st, 2011], Available from http://www.sas.com/.

[30] Sun Microsystems, JAVA, [Online], [Cited May 1st, 2011], Available from http://java.sun.com/.

[31] UBUNTU, Available from http://www.ubuntu.com/.

[32] Van den Berg JP, 1996, *Multiple order pick sequencing in a carousel system: A solvable case of the rural postman*, The Journal of the Operational Research Society, **47(12)**, pp. 1504–1515.

[33] Van den Berg JP & Zijm WHM, 1999, *Models for warehouse management: Classification and examples*, International Journal of Production Economics, **59**, pp. 519–528.

[34] Wallmart, 2010, [Cited February 22nd, 2011], Available from `http://www.walmart.com/`.

[35] Woodall DR, 1972, *Sufficient conditions for cycles in digraphs*, Proceedings London Mathematical Society, **24**, pp. 739–755.

[36] XJ Technologies, AnyLogic 6.5.1, [Online], [Cited Nov 1st, 2011], Available from `http://www.xjtek.com/`.

APPENDIX A

BOSP exact solution formulations

Two additional approaches to the OSP are given in this Appendix. For the first approach the objective function calculated the number of whole cycles needed to pick all the orders. For this formulation subtour constraints were ignored because if a subtour is present in the model it will automatically be penalised with an additional cycle if the subtour does not already ensure the whole completion of a number of cycles. This model has a slight relaxation in that the different subtours may not be able to link together with corresponding start and end locations but will give a lower bound.

In order to formulate this model let

$$x_{ikl} = \begin{cases} 1 & \text{if order } k \text{ starting at bay location } i \text{ is followed by order } l \\ 0 & \text{otherwise,} \end{cases}$$

The following parameters are used in the model. Let

n be the total number of orders,

m be the total number of locations.

$$d'_{ik} = \begin{cases} 1 & \text{if branch order } k \text{ starting at location } i \text{ passes bay location } m \\ 0 & \text{otherwise,} \end{cases}$$

at bay location k,

$$e_{ikj} = \begin{cases} 1 & \text{if branch order } k \text{ starting at bay location } i \text{ is completed at bin location } j \\ 0 & \text{otherwise,} \end{cases}$$

The objective is then to

$$\text{minimise} \quad \sum_{i=1}^{m}\sum_{k=1}^{n}\sum_{l=1}^{n} d'_{ik}x_{ikl} \tag{A.1}$$

subject to

$$\sum_{i=1}^{m}\sum_{k=1}^{n} x_{ikl} = 1 \qquad \forall l, \tag{A.2}$$

$$\sum_{i=1}^{m}\sum_{l=1}^{n} x_{ikl} = 1 \qquad \forall k, \tag{A.3}$$

$$x_{ikl} \leq \sum_{p=1}^{m}\sum_{r=1}^{n} e_{ikp}x_{plr} \qquad \forall i,k,l, \tag{A.4}$$

$$x_{ikl} \in \{0,1\} \qquad \forall i,j,k, \tag{A.5}$$

The objective function (A.1) minimises the number of times a picker is required to pass bay location m and thus minimises the number of cycles required to complete the set of branch orders. Equation sets (A.2) and (A.3) ensure that each order is completed only once. Equation set (A.4) ensures that if two orders follow each other their start and end locations will correspond.

The following model makes use of two separate variable sets, one for sequencing orders and another for determining the starting locations for each order. To model the OSP in this way let

$$x_{ik} = \begin{cases} 1 & \text{if order } k \text{ starts at location } i \\ 0 & \text{otherwise,} \end{cases}$$

$$y_{kl} = \begin{cases} 1 & \text{if order } k \text{ follows branch order } l \\ 0 & \text{otherwise,} \end{cases}$$

The following parameters are used in the model. Let

n	be the total number of orders,
m	be the total number of locations.
$d'_{ik} =$	$\begin{cases} 1 & \text{if order } k \text{ starting at location } i \text{ passes bay location } m \\ 0 & \text{otherwise,} \end{cases}$
$e_{ikj} =$	$\begin{cases} 1 & \text{if order } k \text{ starting at location } i \text{ is completed at location } j \\ 0 & \text{otherwise,} \end{cases}$

The objective is then to

$$\text{minimise} \quad \sum_{i=1}^{m}\sum_{k=1}^{n} d'_{ik}x_{ik} \tag{A.6}$$

subject to

$$\sum_{k=1}^{n} y_{kl} = 1 \qquad \forall l, \tag{A.7}$$

$$\sum_{l=1}^{n} y_{kl} = 1 \qquad \forall k, \tag{A.8}$$

$$x_{ip} \geq -2(1-y_{lp}) + \sum_{j=0}^{m} e_{jli}x_{jl} \qquad \forall i,p,l \text{ where } \sum_{i}^{m} e_{jli} \neq 0, \tag{A.9}$$

$$x_{ip} \leq 1 - y_{lp} \qquad \forall i,p,l \text{ where } \sum_{i}^{m} e_{jli} = 0, \tag{A.10}$$

$$\sum_{i=1}^{m} x_{ik} = 1 \qquad \forall k, \tag{A.11}$$

$$\sum_{k=0}^{n} x_{ik} = 1 \qquad \forall i, \tag{A.12}$$

$$x_{ik} \in \{0,1\} \qquad \forall i,j, \tag{A.13}$$

$$y_{kl} \in \{0,1\} \qquad \forall i,j. \tag{A.14}$$

The objective function (A.6) minimises the number of times a picker is required to pass bay location m and thus minimises the number of cycles required to complete the set of branch orders. Equation sets (A.7) and (A.8) ensures that each order is completed only once. Equation set (A.9) and (A.10) ensures that if two orders follow each other their start and end locations will correspond. Equation sets (A.11) and (A.12) ensure that each order is allocated a single starting location.

APPENDIX B

Simulation figures

This chapter contains some screen shots of the simulation model used for the results validation.

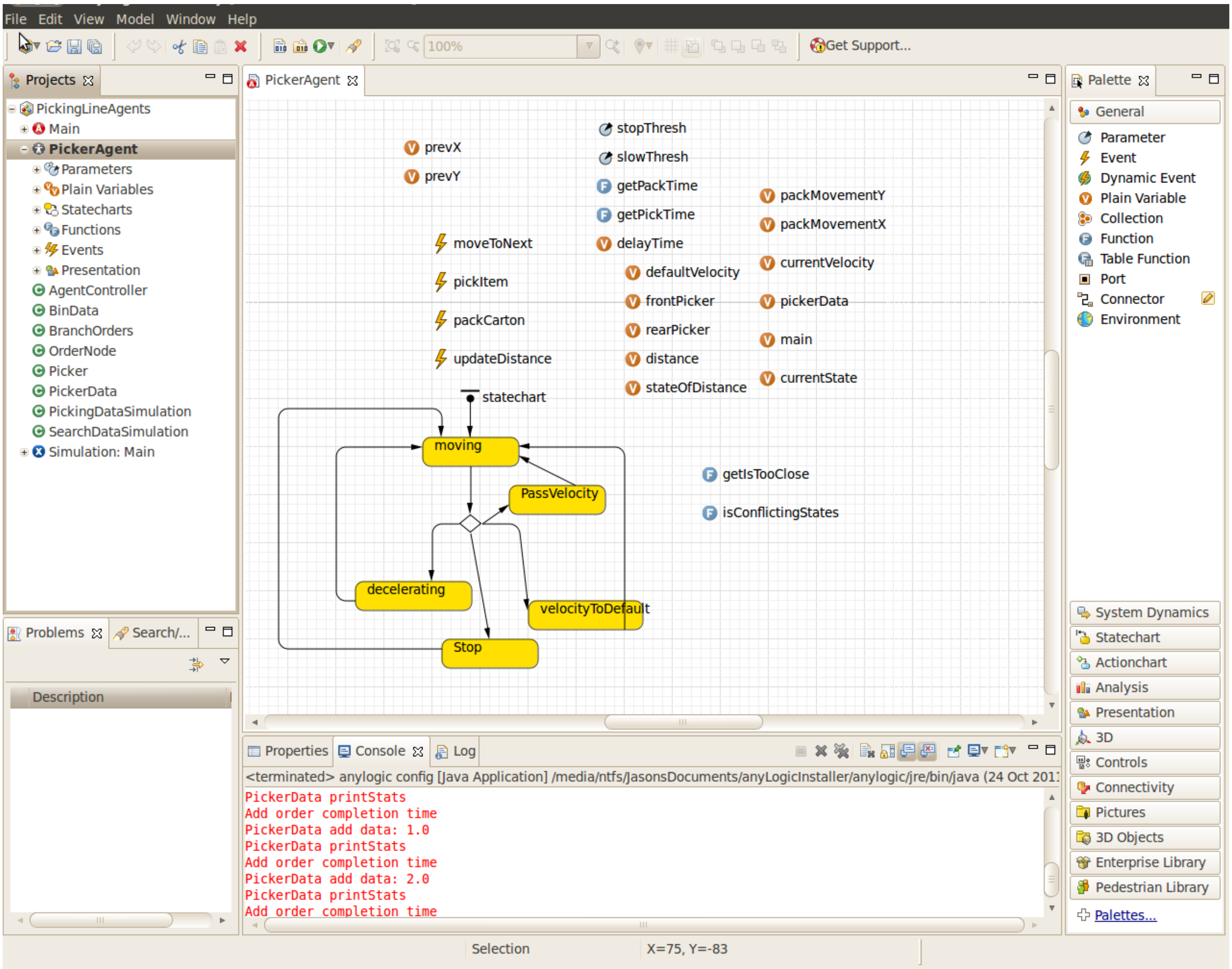


Figure B.1: *A screen shot of the Anylogic development environment used for the simulation.*

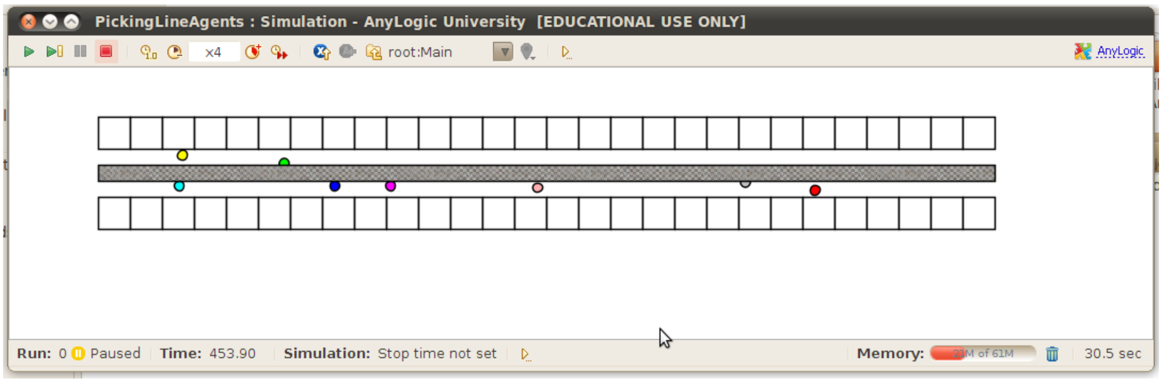


Figure B.2: *A screen shot of a functioning simulation model. The different coloured dots indicate different pickers.*

APPENDIX C

Parameter configurations for ant colony variations

C.1 ASA

Bonferoni grouping	Mean score	α	β
A	1.00186	1	5
A	1.004	1	3
A	1.00409	0	5
A	1.00601	0	3
A	1.00785	0	2.5
A	1.00954	1	2.5
A	1.01036	1	2
A	1.01106	0	2
A	1.01135	1	1.5
A	1.01525	0	1.5
A	1.02107	1	1
A	1.02202	0	1
A	1.02383	1	0.5
A	1.02642	0	0.5
A	1.02792	0	0.1
A	1.03015	0	0
A	1.03046	0	0.3
A	1.03154	1	0
A	1.03167	1	0.1
A	1.03228	1	0.3

Table C.1: *The Bonferoni groupings and mean scores (solution quality relative to the best solution obtained) of the best parameter configuration for the ASA approach for large sized data sets where duplicated SKUs are not present. Elements with the same group within the same class exhibit no significant difference in performance.*

Bonferoni grouping	Mean score	α	β
A	1.0218	1	3
A	1.0224	0	3
A	1.0225	1	2.5
A	1.0234	0	2.5
A	1.0379	1	5
A	1.0383	0	5
A	1.0601	1	2
A	1.0652	0	2
A	1.104	1	1.5
A	1.1067	0	1.5
A	1.1769	0	1
A	1.1793	1	1
A	1.3064	0	0
A	1.3113	1	0.3
A	1.3126	1	0
A	1.3133	0	0.3
A	1.3136	0	0.1
A	1.3172	1	0.1
A	1.3191	0	0.5
A	1.3197	1	0.5

Table C.2: *The Bonferoni groupings and mean scores (solution quality relative to the best solution obtained) of the best parameter configuration for the ASA approach for medium sized data sets where duplicated SKUs are not present. Elements with the same group within the same class exhibit no significant difference in performance.*

Bonferoni grouping	Mean score	α	β
A	1.01429	1	5
A	1.01429	1	3
A	1.01429	0	5
A	1.01429	0	3
A	1.02857	0	2
A	1.02857	1	2
A	1.03333	1	1.5
A	1.03333	0	1.5
A	1.0619	1	1
A	1.0619	0	1
A	1.07262	0	2.5
A	1.07262	1	2.5
A	1.07619	1	0.1
A	1.07619	0	0.3
A	1.07619	1	0.5
A	1.07619	1	0.3
A	1.07619	0	0.5
A	1.07619	0	0.1
A	1.09048	1	0
A	1.09048	0	0

Table C.3: *The Bonferoni groupings and mean scores (solution quality relative to the best solution obtained) of the best parameter configuration for the ASA approach for small sized data sets where duplicated SKUs are not present. Elements with the same group within the same class exhibit no significant difference in performance.*

Bonferroni grouping	Mean score	α	β
A	1.006889	1	5
A	1.01124	1	2.5
A	1.011352	1	3
A	1.017061	0	5
A	1.017067	0	3
A	1.017862	1	2
A	1.018278	1	1.5
A	1.021311	0	2.5
A	1.023257	0	2
A	1.02711	0	1.5
A	1.027422	1	1
A	1.027601	0	1
A	1.029334	1	0.5
A	1.030258	1	0.3
A	1.031214	1	0
A	1.032042	0	0.5
A	1.032788	0	0.1
A	1.033393	0	0.3
A	1.034196	1	0.1
A	1.038051	0	0

Table C.4: *The Bonferoni groupings and mean scores (solution quality relative to the best solution obtained) of the best parameter configuration for the ASA approach for large sized data sets where duplicated SKUs are present. Elements with the same group within the same class exhibit no significant difference in performance.*

Bonferoni grouping	Mean score	α	β
A	1.0092	1	2.5
A	1.0157	0	2.5
A	1.0297	1	3
A	1.0312	0	3
A	1.0322	0	5
A	1.0326	1	5
A	1.0504	0	2
A	1.0556	1	2
A	1.0977	0	1.5
A	1.102	1	1.5
A	1.2383	1	1
A	1.2419	0	1
A	1.3284	0	0.5
A	1.3295	1	0.5
A	1.3614	0	0
A	1.3632	1	0
A	1.371	0	0.1
A	1.3742	0	0.3
A	1.3742	1	0.1
A	1.3743	1	0.3

Table C.5: *The Bonferoni groupings and mean scores (solution quality relative to the best solution obtained) of the best parameter configuration for the ASA approach for medium sized data sets where duplicated SKUs are present. Elements with the same group within the same class exhibit no significant difference in performance.*

Bonferoni grouping	Mean score	α	β
A	1	1	5
A	1.05	0	5
A	1.1222	1	3
A	1.1222	0	3
A	1.1222	1	1.5
A	1.1222	0	1.5
A	1.1222	1	2.5
A	1.1622	0	2.5
A	1.1722	1	2
A	1.1944	1	1
A	1.1944	1	0
A	1.1944	1	0.3
A	1.1944	1	0.1
A	1.1944	0	0.3
A	1.1944	1	0.5
A	1.1944	0	1
A	1.1944	0	0.5
A	1.1944	0	0
A	1.2122	0	2
A	1.2167	0	0.1

Table C.6: *The Bonferoni groupings and mean scores (solution quality relative to the best solution obtained) of the best parameter configuration for the ASA approach for small sized data sets where duplicated SKUs are present. Elements with the same group within the same class exhibit no significant difference in performance.*

C.2 ACC

Bonferoni grouping	Mean score	β
A	1.003086	10
A	1.003842	0.5
A	1.004084	1
A	1.004587	3
A	1.005016	0.1
A	1.005614	5
A	1.005772	2

Table C.7: *The Bonferoni groupings and mean scores (solution quality relative to the best solution obtained) of the best parameter configuration for the ACC cluster sequencing approach for large sized data sets where duplicated SKUs are not present. Elements with the same group within the same class exhibit no significant difference in performance.*

Bonferoni grouping	Mean score	β
A	1.0136	1
A	1.02062	0.1
A	1.02408	10
A	1.02834	5
A	1.03205	3
A	1.03783	0.5
A	1.03894	2

Table C.8: *The Bonferoni groupings and mean scores (solution quality relative to the best solution obtained) of the best parameter configuration for the ACC cluster sequencing approach for medium sized data sets where duplicated SKUs are not present. Elements with the same group within the same class exhibit no significant difference in performance.*

Bonferoni grouping	Mean score	β
A	1	10
A	1	0.5
A	1	3
A	1	1
A	1.01538	2
A	1.01538	5
A	1.01538	0.1

Table C.9: *The Bonferoni groupings and mean scores (solution quality relative to the best solution obtained) of the best parameter configuration for the ACC cluster sequencing approach for small sized data sets where duplicated SKUs are not present. Elements with the same group within the same class exhibit no significant difference in performance.*

Bonferoni grouping	Mean score	β
A	1.01999	3
A	1.02703	5
A	1.03016	1
A	1.03141	10
A	1.03283	2
A	1.03413	0.5
A	1.03882	0.1

Table C.10: *The Bonferoni groupings and mean scores (solution quality relative to the best solution obtained) of the best parameter configuration for the ACC cluster sequencing approach for large sized data sets where duplicated SKUs are present. Elements with the same group within the same class exhibit no significant difference in performance.*

Bonferroni grouping	Mean score	β
A	1.0312	5
A	1.03489	2
A	1.03551	0.5
A	1.03564	10
A	1.03943	3
A	1.04304	1
A	1.04406	0.1

Table C.11: *The Bonferoni groupings and mean scores (solution quality relative to the best solution obtained) of the best parameter configuration for the ACC cluster sequencing approach for medium sized data sets where duplicated SKUs are present. Elements with the same group within the same class exhibit no significant difference in performance.*

Bonferroni grouping	Mean score	β
A	1.02857	10
A	1.02857	0.1
A	1.02857	1
A	1.05	5
A	1.07857	2
A	1.07857	0.5
A	1.07857	3

Table C.12: *The Bonferoni groupings and mean scores (solution quality relative to the best solution obtained) of the best parameter configuration for the ACC cluster sequencing approach for small sized data sets where duplicated SKUs are present. Elements with the same group within the same class exhibit no significant difference in performance.*

APPENDIX D

Parameter testing for clustering variations of the SLP

D.1 MA clustering variation

Bonferoni group	Mean	Maximum cluster size	Number of clusters
A	1.00585	10	16
A	1.006	100	14
A	1.006044	20	16
A	1.006212	15	10
A	1.006363	20	8
A	1.007166	20	10
A	1.007225	10	14
A	1.007301	100	12
A	1.007317	15	12
A	1.007344	10	12
A	1.007624	15	16
A	1.007711	20	12
A	1.007874	10	8
A	1.007934	100	16
A	1.00822	15	8
A	1.008318	15	14
A	1.008337	20	14
A	1.008446	20	6
A	1.008549	100	8
A	1.008921	100	10
A	1.009075	100	6
A	1.009232	10	10
A	1.010249	15	6

Table D.1: *The Bonferoni groupings and mean scores for different combinations of maximal cluster size and number of clusters for the MA clustering variation on large data sets where duplicates are present. Elements with the same group within the same class exhibit no significant difference in performance*
.

Bonferroni group	Mean	Maximum cluster size	Number of clusters
A	1.06821	15	16
A	1.07424	10	16
A	1.07632	100	16
A	1.07768	20	14
A	1.07822	10	12
A	1.07831	10	14
A	1.08274	20	16
A	1.08299	20	12
A	1.0843	100	8
A	1.08432	15	12
A	1.08587	15	14
A	1.08894	100	12
A	1.08917	15	8
A	1.0902	10	10
A	1.09154	15	6
A	1.09387	20	6
A	1.09422	20	8
A	1.09487	100	6
A	1.09518	10	6
A	1.09668	15	10
A	1.0972	10	8
A	1.09753	100	10
A	1.10006	100	14

Table D.2: *The Bonferoni groupings and mean scores for different combinations of maximal cluster size and number of clusters for the MA clustering variation on medium data sets where duplicates are present. Elements with the same group within the same class exhibit no significant difference in performance*
.

Bonferroni group	Mean	Maximum cluster size	Number of clusters
A	1.03077	10	16
A	1.07077	20	6
A	1.07077	20	16
A	1.07077	20	14
A	1.07077	20	8
A	1.07077	20	10
A	1.07077	15	8
A	1.07077	15	6
A	1.07077	15	16
A	1.07077	15	14
A	1.07077	15	12
A	1.07077	15	10
A	1.07077	100	8
A	1.07077	100	6
A	1.07077	100	16
A	1.07077	100	14
A	1.07077	100	12
A	1.07077	100	10
A	1.07077	10	8
A	1.07077	10	6
A	1.07077	20	12
A	1.07077	10	14
A	1.07077	10	12

Table D.3: *The Bonferoni groupings and mean scores for different combinations of maximal cluster size and number of clusters for the MA clustering variation on small data sets where duplicates are present. Elements with the same group within the same class exhibit no significant difference in performance*
.

Bonferroni group	Mean	Maximum cluster size	Number of clusters
A	1.01607	20	16
A	1.01681	15	16
A	1.01687	10	16
A	1.01718	10	12
A	1.01746	20	14
A	1.0194	15	14
A	1.01955	15	12
A	1.01959	100	14
A	1.01986	20	10
A	1.01994	10	14
A	1.02043	15	10
A	1.02066	20	12
A	1.0209	100	12
A	1.02118	100	16
A	1.02138	100	10
A	1.02539	10	10
A	1.03001	20	8
A	1.03064	15	8
A	1.0309	100	8
A	1.03336	10	8
A	1.04017	15	6
A	1.04922	10	6
A	1.04936	100	6

Table D.4: *The Bonferoni groupings and mean scores for different combinations of maximal cluster size and number of clusters for the MA clustering variation on large data sets where duplicates are not present. Elements with the same group within the same class exhibit no significant difference in performance*
.

Bonferroni group	Mean	Maximum cluster size	Number of clusters
A	1.12695	100	14
A	1.12855	20	14
A	1.14121	15	16
A	1.14905	20	16
A	1.15368	15	14
A	1.15766	10	16
A	1.15788	10	14
A	1.16278	100	16
A	1.17023	100	12
A	1.18943	15	12
A	1.19247	20	12
A	1.19495	100	10
A	1.19584	20	10
A	1.19846	10	12
A	1.19915	15	10
A	1.22283	10	10
A	1.22487	15	8
A	1.22957	20	8
A	1.23137	100	8
A	1.23542	15	6
A	1.23663	10	6
A	1.23902	20	6
A	1.24192	10	8

Table D.5: *The Bonferoni groupings and mean scores for different combinations of maximal cluster size and number of clusters for the MA clustering variation on medium data sets where duplicates are not present. Elements with the same group within the same class exhibit no significant difference in performance*

.

Bonferroni group	Mean	Maximum cluster size	Number of clusters
A	1.10714	10	16
A	1.10714	100	16
A	1.15714	20	6
A	1.15714	20	14
A	1.15714	20	8
A	1.15714	20	10
A	1.15714	20	16
A	1.15714	15	6
A	1.15714	15	16
A	1.15714	15	14
A	1.15714	15	12
A	1.15714	15	10
A	1.15714	100	8
A	1.15714	100	6
A	1.15714	15	8
A	1.15714	100	14
A	1.15714	100	12
A	1.15714	100	10
A	1.15714	10	8
A	1.15714	10	6
A	1.15714	20	12
A	1.15714	10	14
A	1.15714	10	12

Table D.6: *The Bonferoni groupings and mean scores for different combinations of maximal cluster size and number of clusters for the MA clustering variation on small data sets where duplicates are not present. Elements with the same group within the same class exhibit no significant difference in performance*
.

D.2 AD clustering variation

Bonferoni group	Mean	Maximum cluster size	Number of clusters
A	1.031629	15	16
A	1.033544	15	12
B, A	1.036062	20	12
B, A	1.03635	20	16
B, A	1.036413	15	14
B, A	1.037903	20	14
C, B, A	1.039061	15	8
C, B, A	1.039605	20	10
D, C, B, A	1.040412	15	6
D, C, B, A	1.040861	15	10
D, C, B, A	1.041866	20	8
D, C, B, A	1.042181	10	10
D, C, B, A	1.043243	10	8
E, D, C, B, A	1.04414	10	16
E, D, C, B, A	1.044598	20	6
E, D, C, B, A	1.047279	10	14
E, D, C, B, A	1.047798	10	12
F, E, D, C, B	1.070724	10	6
F, E, D, C	1.074725	100	16
F, E, D	1.076342	100	14
F, E	1.079952	100	12
F	1.088648	100	10
F	1.100922	100	8

Table D.7: *The Bonferoni groupings and mean scores for different combinations of maximal cluster size and number of clusters for the AD clustering variation on large data sets where duplicates are present. Elements with the same group within the same class exhibit no significant difference in performance*
.

Bonferroni group	Mean	Maximum cluster size	Number of clusters
A	1.3263	15	10
A	1.3374	10	14
A	1.3389	15	14
A	1.3399	10	10
A	1.349	20	10
A	1.3512	15	8
A	1.3519	15	6
A	1.3542	10	16
A	1.3609	20	12
A	1.3616	10	8
A	1.3632	20	14
A	1.3642	20	6
A	1.3646	15	16
A	1.365	15	12
A	1.3654	100	14
A	1.3662	20	8
A	1.3741	100	10
A	1.3746	100	16
A	1.3775	10	12
A	1.3835	20	16
A	1.3837	100	12
A	1.4182	100	8
A	1.42	10	6

Table D.8: *The Bonferoni groupings and mean scores for different combinations of maximal cluster size and number of clusters for the AD clustering variation on medium data sets where duplicates are present. Elements with the same group within the same class exhibit no significant difference in performance*
.

Bonferroni group	Mean	Maximum cluster size	Number of clusters
A	1.2143	15	16
A	1.2143	10	16
A	1.2257	10	6
A	1.2357	15	14
A	1.2357	15	12
A	1.2357	10	14
A	1.2357	10	12
A	1.2357	15	6
A	1.2357	15	10
A	1.2543	10	10
A	1.2757	10	8
A	1.2757	100	16
A	1.2757	15	8
A	1.2757	20	10
A	1.2757	20	14
A	1.2757	100	14
A	1.3043	20	8
A	1.3043	20	6
A	1.3157	100	12
A	1.3157	20	16
A	1.3157	100	10
A	1.3443	100	8
A	1.3443	20	12

Table D.9: *The Bonferoni groupings and mean scores for different combinations of maximal cluster size and number of clusters for the AD clustering variation on small data sets where duplicates are present. Elements with the same group within the same class exhibit no significant difference in performance*
.

Bonferroni group	Mean	Maximum cluster size	Number of clusters
A	1.01363	10	16
A	1.013652	10	14
A	1.014599	10	12
A	1.01469	20	10
A	1.015035	10	10
A	1.015204	20	14
A	1.015351	20	12
A	1.015602	20	6
A	1.015743	20	8
A	1.015892	20	16
A	1.015951	100	10
A	1.016061	10	8
A	1.016187	10	6
A	1.016225	100	8
A	1.016291	100	12
A	1.016487	15	8
A	1.016512	100	6
A	1.016536	100	16
A	1.016604	100	14
A	1.016922	15	10
A	1.01708	15	14
A	1.017163	15	12
A	1.017378	15	6

Table D.10: *The Bonferoni groupings and mean scores for different combinations of maximal cluster size and number of clusters for the AD clustering variation on large data sets where duplicates are not present. Elements with the same group within the same class exhibit no significant difference in performance*
.

Bonferoni group	Mean	Maximum cluster size	Number of clusters
A	1.09495	20	14
A	1.09608	20	16
A	1.09614	20	10
A	1.0967	10	8
A	1.09778	100	16
A	1.09906	20	8
A	1.10025	10	10
A	1.10025	100	10
A	1.10142	100	14
A	1.10222	15	6
A	1.10292	15	14
A	1.1031	100	12
A	1.1044	10	14
A	1.1046	15	8
A	1.10592	15	10
A	1.10772	100	6
A	1.10794	10	12
A	1.10893	15	12
A	1.11054	20	6
A	1.11078	15	16
A	1.11095	100	8
A	1.11133	20	12
A	1.11345	10	6

Table D.11: *The Bonferoni groupings and mean scores for different combinations of maximal cluster size and number of clusters for the AD clustering variations on medium data sets where duplicates are not present. Elements with the same group within the same class exhibit no significant difference in performance*

.

Bonferroni group	Mean	Maximum cluster size	Number of clusters
A	1.04754	100	16
A	1.05459	15	6
A	1.05538	10	12
A	1.05538	10	14
A	1.05538	10	10
A	1.05687	100	6
A	1.05766	20	10
A	1.06279	20	16
A	1.06279	15	12
A	1.06279	100	14
A	1.06359	10	16
A	1.06359	10	6
A	1.06372	10	8
A	1.06507	20	8
A	1.0652	100	8
A	1.06792	15	16
A	1.0702	20	6
A	1.07113	100	10
A	1.07113	15	8
A	1.0734	100	12
A	1.07613	15	10
A	1.07613	15	14
A	1.07613	20	14

Table D.12: *The Bonferoni groupings and mean scores for different combinations of maximal cluster size and number of clusters for the AD clustering variation on small data sets where duplicates are not present. Elements with the same group within the same class exhibit no significant difference in performance*
.

D.3 SA clustering variation

Bonferoni group	Mean	Maximum cluster size	Number of clusters
A	1.02798	10	10
A	1.02953	10	12
A	1.02981	10	16
A	1.03004	10	14
A	1.03196	10	8
A	1.03845	100	14
A	1.04047	100	16
A	1.04053	100	10
A	1.04069	100	12
A	1.04093	20	16
A	1.04151	100	8
A	1.04244	15	8
A	1.0427	15	16
A	1.04275	20	14
A	1.04303	15	6
A	1.04318	15	10
A	1.04357	15	14
A	1.04417	15	12
A	1.04433	100	6
A	1.04622	20	12
A	1.04654	20	10
A	1.04764	20	6
A	1.04787	20	8

Table D.13: *The Bonferoni groupings and mean scores for different combinations of maximal cluster size and number of clusters for the SA clustering variation on large data sets where duplicates are present. Elements with the same group within the same class exhibit no significant difference in performance*
.

Bonferroni group	Mean	Maximum cluster size	Number of clusters
A	1.08659	15	10
A	1.08726	15	12
A	1.08904	15	14
A	1.09367	15	16
A	1.09518	20	16
A	1.09801	10	8
A	1.1006	10	16
A	1.10175	15	6
A	1.10398	100	12
A	1.10581	20	14
A	1.10632	10	10
A	1.1145	10	12
A	1.11462	20	10
A	1.11557	100	10
A	1.1202	100	14
A	1.12177	15	8
A	1.12772	20	8
A	1.13552	10	14
A	1.13746	100	16
A	1.13765	100	6
A	1.14069	20	6
A	1.14155	10	6
A	1.14515	100	8

Table D.14: *The Bonferoni groupings and mean scores for different combinations of maximal cluster size and number of clusters for the SA clustering variation on medium data sets where duplicates are present. Elements with the same group within the same class exhibit no significant difference in performance*
.

Bonferroni group	Mean	Maximum cluster size	Number of clusters
A	1.02857	100	12
A	1.02857	20	6
A	1.02857	20	16
A	1.02857	20	14
A	1.02857	20	12
A	1.02857	10	14
A	1.02857	15	8
A	1.02857	100	14
A	1.02857	15	16
A	1.02857	100	10
A	1.02857	15	12
A	1.02857	15	10
A	1.02857	100	8
A	1.02857	100	6
A	1.02857	100	16
A	1.02857	10	16
A	1.02857	10	8
A	1.02857	10	12
A	1.05714	15	6
A	1.05714	15	14
A	1.07857	10	6
A	1.07857	20	10
A	1.07857	20	8

Table D.15: *The Bonferoni groupings and mean scores for different combinations of maximal cluster size and number of clusters for the SA clustering variation on small data sets where duplicates are present. Elements with the same group within the same class exhibit no significant difference in performance*
.

Bonferroni group	Mean	Maximum cluster size	Number of clusters
A	1.010083	10	6
A	1.011047	10	16
A	1.011149	10	14
A	1.011446	10	12
A	1.011568	15	8
A	1.011805	10	10
A	1.012338	10	8
A	1.012787	15	12
A	1.012928	15	10
A	1.012985	15	6
A	1.013379	20	10
A	1.013657	20	12
A	1.013913	20	8
A	1.014078	20	14
A	1.014153	15	14
A	1.014245	100	10
A	1.014745	20	6
A	1.01492	15	16
A	1.015023	100	6
A	1.015159	20	16
A	1.015218	100	16
A	1.015475	100	14
A	1.01558	100	8

Table D.16: *The Bonferoni groupings and mean scores for different combinations of maximal cluster size and number of clusters for the SA clustering variation on large data sets where duplicates are not present. Elements with the same group within the same class exhibit no significant difference in performance*
.

Bonferoni group	Mean	Maximum cluster size	Number of clusters
A	1.05166	15	6
A	1.0551	10	10
A	1.05639	15	10
A	1.05969	15	14
A	1.06075	15	16
A	1.06144	10	6
A	1.06165	10	14
A	1.06305	10	12
A	1.06324	10	8
A	1.06414	15	8
A	1.06912	10	16
A	1.07252	20	14
A	1.07432	20	10
A	1.0748	15	12
A	1.07577	20	16
A	1.079	20	6
A	1.08398	20	8
A	1.08992	20	12
A	1.09353	100	16
A	1.09727	100	10
A	1.10109	100	12
A	1.10467	100	8
A	1.10861	100	6

Table D.17: *The Bonferoni groupings and mean scores for different combinations of maximal cluster size and number of clusters for the SA clustering variation on medium data sets where duplicates are not present. Elements with the same group within the same class exhibit no significant difference in performance*
.

Bonferroni group	Mean	Maximum cluster size	Number of clusters
A	1.0	15	12
A	1.0	15	8
A	1.0	100	16
A	1.01538	15	6
A	1.03077	100	6
A	1.03077	100	8
A	1.04	15	14
A	1.04	10	6
A	1.04	20	8
A	1.04	20	10
A	1.04	10	16
A	1.04	15	10
A	1.04	20	16
A	1.04	10	14
A	1.04	10	8
A	1.04	10	10
A	1.04	10	12
A	1.05538	20	6
A	1.05538	15	16
A	1.05538	20	14
A	1.05538	100	12
A	1.05538	100	14
A	1.05538	20	12

Table D.18: *The Bonferoni groupings and mean scores for different combinations of maximal cluster size and number of clusters for the SA clustering variation on small data sets where duplicates are not present. Elements with the same group within the same class exhibit no significant difference in performance*
.

D.4 SAD clustering variation

Bonferoni group	Mean	Maximum cluster size	Number of clusters
A	1.022706	10	8
A	1.023766	15	6
A	1.024209	20	8
A	1.024374	20	10
B, A	1.025496	15	8
B, A	1.025643	10	10
B, A	1.025761	10	12
B, A	1.026389	20	12
B, A	1.026398	10	6
B, A	1.026727	20	6
B, A	1.027244	15	10
B, A	1.027658	15	14
B, A	1.028073	10	16
B, A	1.029442	20	16
B, A	1.029551	20	14
B, A	1.030365	15	16
B, A	1.030519	15	12
B, A	1.030654	10	14
B, A	1.043254	100	16
B, A	1.046282	100	14
B, A	1.0478	100	12
B, A	1.051864	100	10
B, A	1.054394	100	8

Table D.19: *The Bonferoni groupings and mean scores for different combinations of maximal cluster size and number of clusters for the SAD clustering variation on large data sets where duplicates are present. Elements with the same group within the same class exhibit no significant difference in performance*
.

Bonferroni group	Mean	Maximum cluster size	Number of clusters
A	1.03653	20	8
A	1.03732	15	12
A	1.04429	15	6
A	1.04523	15	10
A	1.04602	15	16
A	1.04677	20	12
A	1.05004	100	8
A	1.05124	15	14
A	1.05201	100	10
A	1.05208	20	6
A	1.05279	15	8
A	1.05303	20	14
A	1.05315	10	10
A	1.0537	10	8
A	1.05378	10	12
A	1.05478	20	10
A	1.05762	100	6
A	1.05772	10	14
A	1.05789	100	12
A	1.06739	100	14
A	1.06965	100	16
A	1.07376	10	16
A	1.07833	10	6

Table D.20: *The Bonferoni groupings and mean scores for different combinations of maximal cluster size and number of clusters for the SAD clustering variation on medium data sets where duplicates are present. Elements with the same group within the same class exhibit no significant difference in performance*
.

Bonferroni group	Mean	Maximum cluster size	Number of clusters
A	1.0	15	8
A	1.0	20	10
A	1.0	15	6
A	1.02857	20	8
A	1.02857	20	12
A	1.02857	20	6
A	1.02857	20	16
A	1.02857	20	14
A	1.02857	15	16
A	1.02857	15	14
A	1.02857	15	12
A	1.02857	15	10
A	1.02857	100	8
A	1.02857	100	6
A	1.02857	100	16
A	1.02857	100	14
A	1.02857	100	12
A	1.02857	100	10
A	1.02857	10	14
A	1.07857	10	12
A	1.07857	10	16
A	1.10714	10	10
A	1.10714	10	8

Table D.21: *The Bonferroni groupings and mean scores for different combinations of maximal cluster size and number of clusters for the SAD clustering variation on small data sets where duplicates are present. Elements with the same group within the same class exhibit no significant difference in performance*
.

Bonferoni group	Mean	Maximum cluster size	Number of clusters
A	1.003724	10	16
A	1.00393	10	14
A	1.004091	20	10
A	1.004219	10	8
A	1.004265	10	6
A	1.004582	10	10
A	1.004597	20	14
A	1.004917	10	12
A	1.004991	15	8
A	1.005094	15	6
A	1.005165	20	8
A	1.005295	15	10
A	1.00537	20	6
A	1.005753	15	12
A	1.006049	20	16
A	1.006193	20	12
A	1.006761	15	14
A	1.006985	15	16
A	1.008432	100	10
A	1.008764	100	12
A	1.009045	100	8
A	1.009217	100	16
A	1.009651	100	14

Table D.22: *The Bonferoni groupings and mean scores for different combinations of maximal cluster size and number of clusters for the SAD clustering variation on large data sets where duplicates are not present. Elements with the same group within the same class exhibit no significant difference in performance*

.

Bonferoni group	Mean	Maximum cluster size	Number of clusters
A	1.0238	10	8
A	1.02616	10	14
A	1.02704	10	10
A	1.03504	10	12
A	1.03766	10	6
A	1.03836	100	14
A	1.03936	10	16
A	1.0522	100	10
A	1.05317	20	16
A	1.05458	20	8
A	1.05471	100	16
A	1.05517	100	6
A	1.05736	100	12
A	1.05988	20	14
A	1.06134	20	10
A	1.06476	100	8
A	1.06743	15	14
A	1.06747	20	6
A	1.06985	20	12
A	1.07161	15	16
A	1.07204	15	6
A	1.07243	15	8
A	1.07359	15	10

Table D.23: *The Bonferoni groupings and mean scores for different combinations of maximal cluster size and number of clusters for the SAD clustering variation on medium data sets where duplicates are not present. Elements with the same group within the same class exhibit no significant difference in performance*
.

Bonferoni group	Mean	Maximum cluster size	Number of clusters
A	1.01538	15	16
A	1.04	20	6
A	1.04	100	16
A	1.04	20	14
A	1.04	100	8
A	1.04	20	10
A	1.04	20	8
A	1.04	100	10
A	1.04	100	12
A	1.04	100	6
A	1.05538	20	16
A	1.05538	15	6
A	1.05538	15	12
A	1.05538	15	10
A	1.05538	15	8
A	1.05538	100	14
A	1.05538	20	12
A	1.05538	15	14
A	1.05538	10	8
A	1.05538	10	6
A	1.05538	10	16
A	1.05538	10	14
A	1.05538	10	12

Table D.24: *The Bonferoni groupings and mean scores for different combinations of maximal cluster size and number of clusters for the SAD clustering variation on small data sets where duplicates are not present. Elements with the same group within the same class exhibit no significant difference in performance*

.

APPENDIX E

Parameter testing for clustering variations of the SLPCF

E.1 MA clustering variation

Bonferoni group	Mean	Maximum cluster size	Number of clusters
A	1.01541	100	10
B, A	1.01744	10	16
B, A	1.0175	15	10
B, A	1.01817	20	10
B, A	1.02106	10	12
B, A	1.02437	15	14
B, A	1.0249	10	14
B, A	1.02539	20	12
B, A	1.02608	100	12
B, A	1.02773	100	14
B, A	1.02784	15	12
B, A	1.02822	20	16
B, A	1.02836	15	16
B, A	1.02841	10	10
B, A	1.02874	20	14
B, A	1.02885	100	16
B, A	1.03112	10	8
B, A	1.03617	20	8
B, A	1.03668	100	8
B, A	1.03831	15	8
B, A	1.0509	15	6
B, A	1.05403	100	6
B, A	1.05464	20	6

Table E.1: *The Bonferoni groupings and mean scores for different combinations of maximal cluster size and number of clusters for the MA clustering variation on large data sets where duplicates are present. Elements with the same group within the same class exhibit no significant difference in performance*
.

Bonferroni group	Mean	Maximum cluster size	Number of clusters
A	1.01884	20	16
A	1.02655	15	16
A	1.02702	100	16
A	1.03283	15	14
A	1.03966	100	14
A	1.04594	20	14
A	1.0501	10	16
A	1.05467	100	12
A	1.05964	20	12
A	1.06431	10	14
A	1.07542	15	12
A	1.11658	10	12
A	1.13509	15	10
A	1.13618	100	10
A	1.14027	10	10
A	1.14993	20	10
A	1.15416	20	8
A	1.15591	100	8
A	1.15686	15	8
A	1.19893	10	8
A	1.25142	100	6
A	1.25729	15	6
A	1.25942	20	6

Table E.2: *The Bonferoni groupings and mean scores for different combinations of maximal cluster size and number of clusters for the MA clustering variation on medium data sets where duplicates are present. Elements with the same group within the same class exhibit no significant difference in performance*

.

Bonferroni group	Mean	Maximum cluster size	Number of clusters
A	1.0	100	16
A	1.01818	20	16
A	1.05	15	16
A	1.06818	10	16
A	1.10152	10	14
A	1.10818	10	6
A	1.10818	20	6
A	1.10818	15	10
A	1.10818	100	12
A	1.10818	10	8
A	1.12636	100	10
A	1.12636	10	10
A	1.12636	20	8
A	1.12636	20	10
A	1.12636	15	8
A	1.13485	100	14
A	1.13485	15	14
A	1.13485	20	14
A	1.14152	20	12
A	1.14152	15	12
A	1.14152	10	12
A	1.15818	100	6
A	1.15818	100	8

Table E.3: *The Bonferoni groupings and mean scores for different combinations of maximal cluster size and number of clusters for the MA clustering variation on small data sets where duplicates are present. Elements with the same group within the same class exhibit no significant difference in performance*
.

Bonferroni group	Mean	Maximum cluster size	Number of clusters
A	1.005379	100	14
A	1.005576	15	14
A	1.006156	100	16
A	1.006421	15	16
A	1.007148	20	16
A	1.007181	15	10
A	1.007367	10	14
A	1.007897	20	14
A	1.008407	10	16
A	1.008881	100	10
A	1.009433	20	10
A	1.011887	15	12
A	1.012389	15	8
A	1.012881	20	8
A	1.013149	100	8
A	1.01318	10	8
A	1.014984	10	12
A	1.016023	20	12
A	1.016116	10	10
A	1.01612	100	12
A	1.023997	10	6
A	1.025978	15	6
A	1.026084	100	6

Table E.4: *The Bonferoni groupings and mean scores for different combinations of maximal cluster size and number of clusters for the MA clustering variation on large data sets where duplicates are not present. Elements with the same group within the same class exhibit no significant difference in performance*
.

Bonferroni group	Mean	Maximum cluster size	Number of clusters
A	1.01591	20	16
A	1.01691	100	16
A	1.01878	10	16
A	1.019	15	16
A	1.03682	15	14
A	1.03894	20	14
A	1.04404	100	14
A	1.04435	10	14
A	1.09326	100	12
A	1.09359	20	12
A	1.10189	10	12
A	1.10219	15	12
A	1.13079	100	10
A	1.13435	20	10
A	1.13869	15	10
A	1.15055	15	8
A	1.15448	20	8
A	1.15663	100	8
A	1.16688	10	10
A	1.17433	10	8
A	1.19951	10	6
A	1.20999	15	6
A	1.21103	20	6

Table E.5: *The Bonferoni groupings and mean scores for different combinations of maximal cluster size and number of clusters for the MA clustering variation on medium data sets where duplicates are not present. Elements with the same group within the same class exhibit no significant difference in performance*

.

Bonferroni group	Mean	Maximum cluster size	Number of clusters
A	1.0	100	14
A	1.02222	20	14
A	1.03556	20	8
A	1.03556	20	16
A	1.03556	20	12
A	1.03556	100	10
A	1.03556	10	8
A	1.03556	20	10
A	1.03556	100	12
A	1.03556	10	10
A	1.03556	100	8
A	1.03556	10	16
A	1.04889	20	6
A	1.04889	100	6
A	1.05556	15	14
A	1.05556	10	14
A	1.05778	100	16
A	1.06889	15	8
A	1.06889	15	12
A	1.06889	15	10
A	1.06889	10	12
A	1.08222	10	6
A	1.08222	15	6

Table E.6: *The Bonferoni groupings and mean scores for different combinations of maximal cluster size and number of clusters for the MA clustering variation on small data sets where duplicates are not present. Elements with the same group within the same class exhibit no significant difference in performance*
.

E.2 AD clustering variation

Bonferoni group	Mean	Maximum cluster size	Number of clusters
A	1.015211	20	10
A	1.016563	20	8
A	1.016779	20	16
A	1.018584	20	6
A	1.019609	10	8
A	1.019828	10	12
A	1.020191	15	6
A	1.021546	15	8
A	1.021983	15	10
A	1.021983	20	12
A	1.022083	15	14
A	1.02216	10	16
A	1.022185	20	14
A	1.022476	15	12
A	1.023091	10	10
A	1.023506	10	14
A	1.02515	10	6
A	1.026184	15	16
A	1.028034	100	16
A	1.030351	100	12
A	1.03457	100	14
A	1.037358	100	10
A	1.045337	100	8

Table E.7: *The Bonferoni groupings and mean scores for different combinations of maximal cluster size and number of clusters for the AD clustering variation on large data sets where duplicates are present. Elements with the same group within the same class exhibit no significant difference in performance*
.

Bonferroni group	Mean	Maximum cluster size	Number of clusters
A	1.03906	10	10
A	1.03924	20	14
A	1.04132	10	12
A	1.04389	20	12
A	1.04666	20	6
A	1.04701	10	8
A	1.04944	10	16
A	1.05017	20	8
A	1.05325	20	16
A	1.05351	20	10
A	1.05541	10	14
A	1.06633	15	8
A	1.06758	15	16
A	1.06767	15	6
A	1.06969	15	12
A	1.07095	15	14
A	1.0759	15	10
A	1.0825	10	6
A	1.08487	100	16
A	1.09914	100	14
A	1.1208	100	8
A	1.12115	100	10
A	1.12781	100	12

Table E.8: *The Bonferoni groupings and mean scores for different combinations of maximal cluster size and number of clusters for the AD clustering variation on medium data sets where duplicates are present. Elements with the same group within the same class exhibit no significant difference in performance*
.

Bonferroni group	Mean	Maximum cluster size	Number of clusters
A	1.02	20	14
A	1.04	10	14
A	1.06	15	16
A	1.06	20	16
A	1.06	10	12
A	1.06	10	10
A	1.08	20	12
A	1.08	20	10
A	1.08	15	12
A	1.08	15	6
A	1.08	15	8
A	1.08	10	8
A	1.1	20	6
A	1.1	15	14
A	1.1	100	12
A	1.1	15	10
A	1.1	10	6
A	1.1	100	14
A	1.12	10	16
A	1.12	100	10
A	1.13	20	8
A	1.14	100	8
A	1.14	100	6

Table E.9: *The Bonferoni groupings and mean scores for different combinations of maximal cluster size and number of clusters for the AD clustering variation on small data sets where duplicates are present. Elements with the same group within the same class exhibit no significant difference in performance*
.

Bonferroni group	Mean	Maximum cluster size	Number of clusters
A	1.00828	20	8
A	1.00841	20	6
A	1.00888	20	12
A	1.00913	20	14
A	1.01145	10	8
A	1.01221	10	12
A	1.01292	20	16
A	1.01296	10	6
A	1.01437	10	10
A	1.01557	15	12
A	1.01588	20	10
A	1.01668	10	14
A	1.01674	15	10
A	1.0169	15	8
A	1.01698	10	16
A	1.01743	15	6
A	1.01794	15	16
A	1.01872	15	14
A	1.02697	100	16
A	1.02926	100	6
A	1.02948	100	8
A	1.02954	100	10
A	1.03038	100	14

Table E.10: *The Bonferoni groupings and mean scores for different combinations of maximal cluster size and number of clusters for the AD clustering variation on large data sets where duplicates are not present. Elements with the same group within the same class exhibit no significant difference in performance*
.

Bonferoni group	Mean	Maximum cluster size	Number of clusters
A	1.02889	20	16
A	1.03096	20	6
A	1.03245	20	10
A	1.04005	20	8
A	1.04166	10	10
A	1.04597	15	10
A	1.04873	20	12
A	1.05242	10	14
A	1.05346	10	8
A	1.05518	15	8
A	1.05526	15	6
A	1.05542	20	14
A	1.05625	10	16
A	1.05683	15	12
A	1.06382	15	14
A	1.06537	10	12
A	1.07038	10	6
A	1.08219	15	16
A	1.10196	100	16
A	1.11861	100	14
A	1.12561	100	6
A	1.12575	100	10
A	1.12884	100	12

Table E.11: *The Bonferoni groupings and mean scores for different combinations of maximal cluster size and number of clusters for the AD clustering variation on medium data sets where duplicates are not present. Elements with the same group within the same class exhibit no significant difference in performance*

.

Bonferroni group	Mean	Maximum cluster size	Number of clusters
A	1.04762	15	16
A	1.04762	15	10
A	1.05833	20	14
A	1.06508	100	6
A	1.06984	10	16
A	1.06984	10	14
A	1.07619	15	8
A	1.07619	10	12
A	1.07619	15	14
A	1.08056	100	12
A	1.08056	100	16
A	1.08373	20	10
A	1.0873	100	8
A	1.09206	15	12
A	1.09484	100	14
A	1.09484	20	16
A	1.10278	20	8
A	1.10635	100	10
A	1.10913	10	10
A	1.11706	20	6
A	1.11706	20	12
A	1.12659	10	6
A	1.13016	15	6

Table E.12: *The Bonferoni groupings and mean scores for different combinations of maximal cluster size and number of clusters for the AD clustering variation on small data sets where duplicates are not present. Elements with the same group within the same class exhibit no significant difference in performance*
.

E.3 SA clustering variation

Bonferoni group	Mean	Maximum cluster size	Number of clusters
A	1.010652	20	16
A	1.011028	20	14
A	1.013472	20	12
A	1.014442	20	10
A	1.014753	20	8
A	1.014953	15	12
A	1.01812	100	16
A	1.019006	15	8
A	1.019099	15	14
A	1.019862	20	6
A	1.019947	100	10
A	1.020321	15	16
A	1.020363	15	10
A	1.0204	10	14
B, A	1.020832	100	12
B, A	1.022019	10	16
B, A	1.022187	10	12
B, A	1.022529	100	14
B, A	1.022911	100	8
B, A	1.024869	10	10
B, A	1.026518	100	6
B, A	1.030768	15	6
B, A	1.034588	10	8

Table E.13: *The Bonferoni groupings and mean scores for different combinations of maximal cluster size and number of clusters for the SA clustering variation on large data sets where duplicates are present. Elements with the same group within the same class exhibit no significant difference in performance*
.

Bonferroni group	Mean	Maximum cluster size	Number of clusters
A	1.02882	15	8
A	1.03726	15	6
A	1.05159	15	10
A	1.06113	15	12
A	1.07101	10	10
A	1.07918	15	16
A	1.08013	15	14
A	1.08168	10	12
A	1.12108	10	8
A	1.12546	20	12
A	1.12575	20	6
A	1.12576	10	14
A	1.13	20	10
A	1.13061	20	14
A	1.13672	10	16
A	1.14541	100	8
A	1.15108	20	16
A	1.15164	20	8
A	1.15867	100	12
A	1.17572	100	10
A	1.17655	100	16
A	1.18355	100	14
A	1.23517	100	6

Table E.14: *The Bonferoni groupings and mean scores for different combinations of maximal cluster size and number of clusters for the SA clustering variation on medium data sets where duplicates are present. Elements with the same group within the same class exhibit no significant difference in performance*
.

Bonferroni group	Mean	Maximum cluster size	Number of clusters
A	1.0	15	16
A	1.0	15	14
A	1.0	10	8
A	1.0	15	8
A	1.0	15	12
A	1.02222	10	16
A	1.02222	10	12
A	1.05	15	10
A	1.05	10	14
A	1.05	15	6
A	1.05	10	10
A	1.07222	10	6
A	1.08	20	12
A	1.08	20	14
A	1.10222	20	16
A	1.11222	100	12
A	1.11222	100	16
A	1.13444	20	6
A	1.15222	100	6
A	1.15222	20	10
A	1.15222	100	8
A	1.15222	100	14
A	1.17444	20	8

Table E.15: *The Bonferoni groupings and mean scores for different combinations of maximal cluster size and number of clusters for the SA clustering variation on small data sets where duplicates are present. Elements with the same group within the same class exhibit no significant difference in performance*
.

Bonferroni group	Mean	Maximum cluster size	Number of clusters
A	1.003066	10	14
A	1.004065	10	10
A	1.004104	10	16
A	1.004325	10	12
A	1.005515	10	8
A	1.005874	10	6
A	1.009838	15	10
A	1.009851	15	14
A	1.010038	15	8
A	1.010817	15	12
A	1.010991	15	16
A	1.013728	15	6
A	1.018102	100	12
A	1.020001	100	6
A	1.020064	20	8
A	1.020148	20	14
A	1.020576	100	10
A	1.020818	100	16
A	1.020909	20	6
A	1.020961	20	12
A	1.021014	20	16
A	1.021118	100	14
A	1.021954	100	8

Table E.16: *The Bonferoni groupings and mean scores for different combinations of maximal cluster size and number of clusters for the SA clustering variation on large data sets where duplicates are not present. Elements with the same group within the same class exhibit no significant difference in performance*
.

Bonferoni group	Mean	Maximum cluster size	Number of clusters
A	1.01252	10	10
A	1.03602	10	12
A	1.04558	10	14
A	1.04761	10	16
A	1.04875	15	8
A	1.05272	15	10
A	1.05324	15	6
A	1.05508	10	8
A	1.05541	15	16
A	1.05675	15	12
A	1.06054	15	14
A	1.10166	20	16
A	1.10368	20	12
A	1.10483	20	8
A	1.10644	10	6
A	1.10977	20	6
A	1.11071	20	10
A	1.11244	20	14
A	1.20841	100	14
A	1.21529	100	12
A	1.21941	100	16
A	1.23435	100	10
A	1.23475	100	8

Table E.17: *The Bonferoni groupings and mean scores for different combinations of maximal cluster size and number of clusters for the SA clustering variation on medium data sets where duplicates are not present. Elements with the same group within the same class exhibit no significant difference in performance*
.

Bonferroni group	Mean	Maximum cluster size	Number of clusters
A	1.0	20	12
A	1.0	20	10
A	1.0	20	16
A	1.0	20	14
A	1.0	10	16
A	1.0	10	14
A	1.0	10	6
A	1.01538	15	8
A	1.01538	20	8
A	1.01538	15	14
A	1.01538	100	16
A	1.01538	15	6
A	1.01538	15	16
A	1.01538	10	10
A	1.01538	10	12
A	1.01538	15	10
A	1.02222	20	6
A	1.03077	100	12
A	1.03077	10	8
A	1.03077	100	10
A	1.04615	100	8
A	1.04615	100	14
A	1.04872	15	12

Table E.18: *The Bonferoni groupings and mean scores for different combinations of maximal cluster size and number of clusters for the SA clustering variation on small data sets where duplicates are not present. Elements with the same group within the same class exhibit no significant difference in performance*
.

E.4 SAD clustering variation

Bonferoni group	Mean	Maximum cluster size	Number of clusters
A	1.010609	15	12
A	1.011101	15	10
A	1.011311	15	8
A	1.011352	15	6
A	1.014699	15	16
A	1.016454	10	8
A	1.016645	20	16
A	1.016657	10	12
A	1.017773	20	14
A	1.01803	15	14
A	1.020792	10	10
A	1.021622	20	10
A	1.021983	20	12
A	1.023001	10	6
A	1.023043	20	8
A	1.024057	10	14
B, A	1.024398	20	6
B, A	1.027541	10	16
C, B	1.056338	100	16
C	1.061809	100	14
C	1.0636	100	8
C	1.064583	100	10
C	1.06538	100	6

Table E.19: *The Bonferoni groupings and mean scores for different combinations of maximal cluster size and number of clusters for the SAD clustering variation on large data sets where duplicates are present. Elements with the same group within the same class exhibit no significant difference in performance*
.

Bonferroni group	Mean	Maximum cluster size	Number of clusters
A	1.0326	15	6
A	1.0487	15	8
A	1.0652	15	12
A	1.0771	15	16
A	1.0775	15	10
A	1.0788	20	10
A	1.0801	20	6
A	1.0844	20	8
A	1.0887	20	16
A	1.092	10	10
A	1.0944	15	14
A	1.0971	10	14
A	1.0972	20	14
A	1.0981	20	12
A	1.1	10	12
A	1.1044	10	8
A	1.1118	10	16
A	1.1806	10	6
A	1.3385	100	14
A	1.3416	100	16
A	1.3481	100	12
A	1.3631	100	6
A	1.3702	100	8

Table E.20: *The Bonferroni groupings and mean scores for different combinations of maximal cluster size and number of clusters for the SAD clustering variation on medium data sets where duplicates are present. Elements with the same group within the same class exhibit no significant difference in performance*
.

Bonferoni group	Mean	Maximum cluster size	Number of clusters
A	1.0	15	16
A	1.0	15	6
A	1.0	15	12
B, A	1.025	15	14
B, A	1.025	15	8
B, A	1.025	20	14
B, A	1.025	10	8
B, A	1.025	15	10
B, A	1.04	20	16
B, A	1.05	20	6
B, A	1.05	10	16
B, A	1.05	10	12
C, B, A	1.075	20	10
C, B, A	1.075	10	14
C, B, A	1.075	20	8
C, B, A	1.075	20	12
C, B, A	1.1	10	10
C, B, A	1.1	10	6
C, B, A	1.115	100	16
C, B, A	1.115	100	14
C, B, A	1.205	100	12
C, B	1.265	100	8
C	1.315	100	10

Table E.21: *The Bonferoni groupings and mean scores for different combinations of maximal cluster size and number of clusters for the SAD clustering variation on small data sets where duplicates are present. Elements with the same group within the same class exhibit no significant difference in performance*
.

Bonferoni group	Mean	Maximum cluster size	Number of clusters
A	1.00754	10	8
A	1.00765	10	10
A	1.00913	10	16
A	1.00982	10	14
A	1.00984	10	6
A	1.01021	10	12
A	1.01705	15	16
A	1.0178	15	12
A	1.01841	20	10
A	1.01841	15	8
A	1.01846	20	14
A	1.01856	20	12
A	1.01857	20	6
A	1.0187	15	6
A	1.01872	15	14
A	1.01894	15	10
A	1.02056	20	16
A	1.0212	20	8
A	1.03811	100	16
A	1.05095	100	10
A	1.05101	100	6
A	1.05128	100	8
A	1.05157	100	14

Table E.22: *The Bonferoni groupings and mean scores for different combinations of maximal cluster size and number of clusters for the SAD clustering variation on large data sets where duplicates are not present. Elements with the same group within the same class exhibit no significant difference in performance*

.

Bonferoni group	Mean	Maximum cluster size	Number of clusters
A	1.02854	10	8
A	1.06677	10	10
A	1.07242	10	14
A	1.07562	15	14
A	1.07628	15	12
A	1.07676	20	6
A	1.07718	15	10
A	1.07996	15	8
A	1.08043	20	14
A	1.08303	15	6
A	1.08765	10	12
A	1.08963	20	8
A	1.09345	20	12
A	1.09441	20	10
A	1.10206	10	16
A	1.1131	10	6
A	1.11407	20	16
A	1.12248	100	12
A	1.12435	15	16
A	1.15243	100	16
A	1.16111	100	14
A	1.18036	100	10
A	1.19291	100	8

Table E.23: *The Bonferoni groupings and mean scores for different combinations of maximal cluster size and number of clusters for the SAD clustering variation on medium data sets where duplicates are not present. Elements with the same group within the same class exhibit no significant difference in performance*
.

Bonferoni group	Mean	Maximum cluster size	Number of clusters
A	1.0	15	16
A	1.0	15	14
A	1.0	20	16
A	1.0	20	14
A	1.0	10	16
A	1.0	10	14
A	1.0	15	8
A	1.0	15	6
A	1.0	15	12
A	1.0	15	10
A	1.0	10	12
A	1.0	100	14
A	1.01429	20	8
A	1.01429	20	6
A	1.01429	100	12
A	1.01429	100	10
A	1.01429	10	8
A	1.01429	10	6
A	1.01429	100	16
A	1.01429	10	10
A	1.02857	20	12
A	1.02857	20	10
A	1.04286	100	6

Table E.24: *The Bonferoni groupings and mean scores for different combinations of maximal cluster size and number of clusters for the SAD clustering variation on small data sets where duplicates are not present. Elements with the same group within the same class exhibit no significant difference in performance*

.