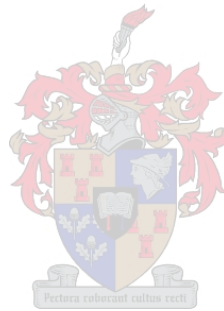


Decision Support for Fresh Produce Replenishment Order Schedules in a Retail Outlet

Janneke Lötter



Thesis presented in partial fulfilment of the requirements for the degree of
Master of (Industrial) Engineering
in the Faculty of Engineering at Stellenbosch University

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.

Date: March 1, 2017

Abstract

Retailers are faced with complex periodic decisions related to replenishment orders for fresh produce and the allocation of shelf space to these products. The complexity of these decisions may be attributed to the short shelf lives of fresh produce, a variety of operational constraints and a limitation on the amount of shelf space available. Since retailers may gain a competitive advantage through the efficient management of shelf space in their fresh produce departments, an objective methodology is required to provide decision support for fresh produce replenishment order decisions. This is especially important for retail outlets with little or no backroom storage space for their fresh produce, where all product units that arrive at the store have to be displayed on the shelves immediately.

A mathematical model is proposed in this thesis in support of the aforementioned fresh produce replenishment order decisions of retailers. The aim of the model is to satisfy forecasted product demand over a prescribed decision period as closely as possible, while maximising profit and minimising product waste — all subject to a fixed amount of available shelf space. Two solution approaches are designed for solving this model which takes the form of a mixed integer programming problem. An exact solution approach is implemented in **CPLEX** for solving the model in the context of small, hypothetical problem instances. Due to the computational complexity of the exact solution approach, a (metaheuristic) solution approach, based on the method of simulated annealing, is also established by which larger, realistic instances of the model can be solved much quicker, albeit approximately.

The mathematical model and the approximate solution approach are embedded in a computerised decision support system concept demonstrator so as to provide managers of retail outlets with a practical tool able to recommend fresh produce replenishment order schedules in line with the aforementioned retail objectives. A real-life case study is performed, involving 72 products in the ambient section of the fresh produce department within a retail outlet in Grassy Park, Cape Town, so as to demonstrate the practicality of the decision support system and illustrate the high quality of its recommendations. Upon comparing the replenishment order recommendation of the system with that actually employed at the outlet, it is found that the system yields substantially superior results.

Uittreksel

Kleinhandelaars staar moeilike besluite rakende aanvullingsbestellings van varsprodukte en die toewysing van rakspasie aan hierdie produkte in die gesig. Die kompleksiteit van hierdie besluite kan toegeskryf word aan die kort rakleef tyd van hierdie produkte, verskeie operasionele beperkings en beperkte beskikbare rakspasie. Aangesien kleinhandelaars 'n kompeterende voordeel mag trek uit die doeltreffende bestuur van rakspasie in hul varsprodukafdelings, word 'n objektiewe metodologie vereis waarvolgens hulle met besluitsteun in terme van varsproduk aanvullingsbesluite bedien kan word. Hierdie vereiste is veral belangrik in gevalle waar kleinhandelwinkels beperkte of geen stoorkamerspasie vir varsprodukte tot hul beskikking het nie, en waar al hierdie produkte dus dadelik op die rakke ten toon gestel moet word.

'n Wiskundige model word in hierdie tesis daargestel waarvolgens steun in terme van die bogenoemde varsproduk aanvullingsbesluite aan kleinhandelaars gebied kan word. Die doel van die model is om sover moontlik aan vooruitgeskatte aanvraag vir hierdie produkte oor 'n voorgeskrewe beslissingsperiode te voldoen, terwyl wins gemaksimeer word en produkvermorsing geminimeer word — alles onderhewig aan 'n vaste hoeveelheid beskikbare rakspasie. Twee oplossingsbenaderings word ontwerp waarvolgens hierdie model, wat die vorm van 'n gemengde heeltallige programmeringsprobleem aanneem, opgelos kan word. 'n Eksakte oplossingstechniek word in CPLEX geïmplementeer waarmee klein, hipotetiese probleemgevalle opgelos kan word. As gevolg van die berekeningskompleksiteit van die eksakte oplossingsbenadering word 'n (metaheuristiese) oplossingstechniek gebaseer op die metode van gesimuleerde tempering ook daargetel waarmee groter, realistiese gevalle van die probleem vinniger, maar benaderd, opgelos kan word.

Die wiskundige model en die benaderde oplossingstechniek word in 'n gerekenariseerde besluitsteun-konsepdeemonstrator ingesluit om sodoende kleinhandelaars van 'n praktiese hulpmiddel te voorsien wat daartoe in staat is om aanbevelings in terme van varsproduk aanvullingsbesluite volgens die bogenoemde kleinhandeldoelwitte te maak. 'n Werklike gevallestudie word uitgevoer waarin 72 produkte wat in die kamertemperatuur-gedeelte van die varsprodukafdeling van 'n kleinhandelwinkel in Grassy Park, Kaapstad in ag geneem word om sodoende die praktiese nut van die besluitsteunstelsel en die hoë kwaliteit van die aanbevelings wat die stelsel genereer, te illustreer. Deur die aanvullingskediule wat die stelsel aanbeveel met dié wat werklik by die winkel gevolg is, te vergelyk, word daar bevind dat die stelsel noemenswaardige beter resultate lewer.

Acknowledgements

The author wishes to acknowledge the following people and institutions for their various contributions towards the completion of this work:

- Friends and family for their prayers, encouragement and support. I could not have done this in my own strength. To me, this thesis will forever be a testimony of God's grace and provision.
- Professor van Vuuren, my supervisor, for his wisdom, insight, knowledge, guidance and continuous feedback throughout this project. He facilitated an amazing learning opportunity and I appreciate all his effort.
- The SUnORE research group, headed by Professor van Vuuren, for funding provided in support of my masters degree and for the privilege of using the exclusive and excellent facilities in the SUnORE office.
- Ruellyn Snyman, the industry partner, for her insight into and practical knowledge of the retail world, her enthusiasm towards this project and her willingness to help until the very end.
- Anglo American for funding provided in support of my masters degree.
- Renier Steynberg, my soon-to-be husband, for being the best supporter I could ask for.
- Fellow SUnORE students during the past two years for an interest in my project, offering advice and help, joyous times together and inspiring me with their work.

Table of Contents

Abstract	iii
Uittreksel	v
Acknowledgements	vii
List of Reserved Symbols	xiii
List of Acronyms	xv
List of Figures	xvii
List of Tables	xix
List of Algorithms	xxi
1 Introduction	1
1.1 Background	1
1.2 Project aim and scope	2
1.3 Thesis objectives	3
1.4 Thesis organisation	4
2 Literature review	7
2.1 The retail industry	7
2.1.1 Basic retail definitions	8
2.1.2 Operations at a retail outlet	9
2.1.3 Customer behaviour in a retail outlet	10
2.1.4 The South African retail industry	11
2.2 Fresh produce	12
2.2.1 Perishable products	12
2.2.2 Food waste	14

2.3	Inventory management and retail shelf space allocation	16
2.3.1	Product assortment	16
2.3.2	Price planning	18
2.3.3	Inventory management	19
2.3.4	Shelf space allocation	24
2.3.5	Integration of related literature	27
2.4	Solution techniques for combinatorial optimisation problems	28
2.4.1	Exact solution techniques	29
2.4.2	Approximate solution techniques	30
2.5	Model validation techniques	33
2.6	Decision support systems	35
2.7	Chapter summary	36
3	Mathematical model formulation	37
3.1	Model assumptions and considerations	38
3.2	The objective function	38
3.3	The shelf space constraint	39
3.4	Ensuring demand satisfaction	40
3.5	Modelling conservation of stock	40
3.6	Modelling waste	40
3.7	Chapter summary	42
4	Model solution by CPLEX	45
4.1	Solving the mathematical model exactly	45
4.2	Product data for hypothetical test instances	46
4.3	Numerical results	48
4.3.1	Solution for the first product in Table 4.1	48
4.3.2	Solution for the first two products in Table 4.1	51
4.3.3	Solution for the first three products in Table 4.1	51
4.3.4	Appraisal of the numerical results	57
4.4	Solving the mathematical model approximately	58
4.4.1	Imposing a solution time limit	58
4.4.2	A problem-specific heuristic based on LP relaxation	60
4.5	Chapter summary	63

5	Model solution by simulated annealing	65
5.1	Algorithm development	65
5.1.1	Constraint handling	66
5.1.2	Construction of an initial solution	66
5.1.3	The move operator	69
5.1.4	Accepting a neighbouring solution	69
5.1.5	The initial temperature	70
5.1.6	The cooling and reheating schedules	70
5.1.7	Search termination criteria	71
5.1.8	Algorithm pseudocode	71
5.1.9	Implementation details	73
5.2	Validation of the approach	73
5.3	Solving a larger problem instance	76
5.4	Chapter summary	77
6	Decision support system	79
6.1	Decision support system architecture	79
6.2	Decision support system implementation	80
6.3	Chapter summary	86
7	A case study	87
7.1	Background	88
7.2	Data preparation	88
7.3	Algorithmic parameter evaluation experimental design	89
7.4	Numerical results	96
7.4.1	Benchmarks	96
7.4.2	Results obtained during the first nine search runs	96
7.4.3	Results obtained during the subsequent twenty six search runs	97
7.4.4	Additional results	98
7.5	Discussion	100
7.6	Chapter summary	104
8	Conclusion	107
8.1	Thesis summary	107
8.2	Appraisal of contributions	109
8.2.1	Appraisal of the modelling framework	109

8.2.2 Appraisal of the case study	110
8.3 Suggestions for future work	111
A CPLEX source code	123
B Simulated annealing algorithm source code	125
C Input data for the hypothetical problem	131
D Additional data for the case study	139

List of Reserved Symbols

Symbol	Meaning
<i>Indices</i>	
i	Index used to reference fresh produce product
j	Index used to reference days in the decision period
<i>Variables</i>	
q_{ij}	Order quantity of product i on day j
r_{ij}	Starting inventory of product i on day j
w_{ij}	Waste of product i discarded at the end of day j
μ_{ij}	Overachievement associated with demand satisfaction of product i on day j
σ_{ij}	Underachievement associated with demand satisfaction of product i on day j
<i>Parameters</i>	
a_i	Unit acquisition cost of product i
b_i	Stacking factor of product i
d_{ij}	Forecasted demand for product i on day j
e_i	Unit display space required by product i
k_i	Pack size of product i
l_i	Lead time of product i
p_i	Unit selling price of product i
s_i	Safety stock required for product i
S	Total amount of shelf space available in the fresh produce department
t_i	Shelf life of product i

List of Acronyms

DSS	Decision Support System
EOQ	Economic Order Quantity
IDE	Integrated Development Environment
IP	Integer Programming
LP	Linear Programming
MIP	Mixed Integer Programming
RFID	Radio-Frequency IDentification
SA	Simulated Annealing
SKU	Stock Keeping Unit
UI	User Interface

List of Figures

1.1	Fresh produce department welcome to customers	2
1.2	Diagram of thesis organisation	4
2.1	Inventory level of basic deterministic continuous review inventory system	21
2.2	Inventory level of inventory system with lead time and safety stock	22
2.3	A classification structure of optimisation methods	29
4.1	Product unit flow summary for first hypothetical product	50
4.2	Daily shelf space utilisation for first two hypothetical products	53
4.3	Product unit flow summary for first two hypothetical products	53
4.4	CPLEX output graphs showing statistics	55
4.5	Daily shelf space utilisation for first three hypothetical products	56
4.6	Product unit flow summary for first three hypothetical products	56
5.1	Accepted solutions when SA applied to larger problem instance	76
5.2	System temperature when SA applied to larger problem instance	77
6.1	DSS architecture	80
6.2	Main screen displayed when UI is launched	81
6.3	Main UI screen allows user to upload input files	82
6.4	Second tab of UI for calculating initial temperature and order schedule	82
6.5	Calculated initial temperature is displayed	83
6.6	User specified 5 000 iterations as limit	83
6.7	Best replenishment order schedule found is displayed	84
6.8	Cost function values of accepted solutions are presented	84
6.9	Output in Excel tables	85
6.10	Three additional UI features	85
7.1	Normalised results of first nine runs in parameter evaluation experiment	97

7.2	Normalised results of next 26 runs in parameter evaluation experiment	98
A.1	CPLEX source code .dat file	123
A.2	CPLEX source code .mod file — Part 1	124
A.3	CPLEX source code .mod file — Part 2	124
B.1	SA source code — Part 1	126
B.2	SA source code — Part 2	127
B.3	SA source code — Part 3	128
B.4	SA source code — Part 4	129

List of Tables

2.1	Ethylene producing and ethylene-sensitive fresh produce	15
3.1	Product characteristics and daily values for waste calculation example	41
3.2	Summary of model indices, parameters and variables	43
4.1	Product characteristics for hypothetical problem instances	47
4.2	Product demands for extended decision period (first three hypothetical products)	47
4.3	CPLEX replenishment order schedule for first hypothetical product	49
4.4	CPLEX inventory values for first hypothetical product	49
4.5	CPLEX waste values for first hypothetical product	49
4.6	CPLEX underachievement values for first hypothetical product	49
4.7	CPLEX overachievement values for first hypothetical product	49
4.8	CPLEX summary of solutions for first hypothetical product	49
4.9	CPLEX replenishment order schedule for first two hypothetical products	52
4.10	CPLEX inventory values for first two hypothetical products	52
4.11	CPLEX waste values for first two hypothetical products	52
4.12	CPLEX summary of solutions for first two hypothetical products	52
4.13	CPLEX summary of solutions for first three hypothetical products	54
4.14	CPLEX summary of solutions for first four hypothetical products	59
4.15	CPLEX summary of solutions for first five hypothetical products	60
4.16	Excerpt of CPLEX replenishment order schedule with LP relaxation	61
4.17	Excerpt of CPLEX replenishment order schedule with Rounding heuristic 1	61
4.18	Excerpt of CPLEX replenishment order schedule with Rounding heuristic 2	62
4.19	Excerpt of CPLEX replenishment order schedule with Rounding heuristic 3	62
4.20	Comparison of three rounding heuristic solutions and MIP solution	63
5.1	Demand for constructing an initial solution	68
5.2	Theoretical, fractional order quantities for constructing an initial solution	68

5.3	Rounded order quantities for constructing an initial solution	69
5.4	Example of a constructed initial solution	69
5.5	Product characteristics for hypothetical problem instances (repeated)	74
5.6	Shelf space values for nine problem instances solved as validation of SA approach	74
5.7	Comparison of SA and CPLEX cost function values for nine problem instances .	75
7.1	Characteristics of 72 products in case study problem	90
7.2	Three values assigned to selected algorithmic parameters	94
7.3	Parameter value combinations for parameter evaluation experiment	95
7.4	Cost function value breakdown for three additional runs	99
7.5	Cost function value breakdown for retailer's replenishment order schedule	100
7.6	Replenishment order schedule from best combination of SA parameter values . .	101
7.7	Comparison of SA replenishment order schedule and retailer's schedule	105
C.1	Product demand for extended decision period (all ten hypothetical products) . .	132
C.2	Historical product demand (all ten hypothetical products)	133
C.3	Historical replenishment order schedule (all ten hypothetical products)	134
C.4	Historical inventory values (all ten hypothetical products)	135
C.5	Historical underachievement values (all ten hypothetical products)	136
C.6	Historical overachievement values (all ten hypothetical products)	137
C.7	Historical waste values (all ten hypothetical products)	138
D.1	Case study product demand	140
D.2	Case study historical demand	141
D.3	Case study historical replenishment order schedule	143
D.4	Case study historical inventory values	144
D.5	Case study historical waste values	146
D.6	Case study initial solution	147
D.7	Replenishment order schedule actually employed in case study retail outlet . . .	149

List of Algorithms

2.1	General simulated annealing algorithm	32
5.1	Simulated annealing algorithm for replenishment order schedules	72

CHAPTER 1

Introduction

Contents

1.1	Background	1
1.2	Project aim and scope	2
1.3	Thesis objectives	3
1.4	Thesis organisation	4

1.1 Background

Upon entering a local supermarket to do grocery shopping, a customer is typically welcomed by a colourful display of flowers, the aroma of freshly baked bread or a careful arrangement of fresh fruit and vegetables, as illustrated in Figure 1.1. It is no coincidence that these departments are often found near the entrance of a retail outlet [84]. Retailers are persuading customers to spend as much time and money in their retail outlets as possible, which is the reason for incorporating decision support based on mathematical models in the layout of their stores, the management of their products and the allocation of retail shelf space.

The retail sector is known to be highly competitive. Retailers are required to decide on the physical layout of the product displays in their stores, to manage their product inventories, to collaborate with suppliers and to take the desires and behaviour of their customers into account — all in an effort to maximise their profits and increase their competitive advantage.

Fresh produce, which refers to fruit and vegetable products in this thesis, plays a vital role in the profitability of a retail outlet. The freshness and colours associated with these products contribute to in-store customer satisfaction. Fresh produce is generally categorised as a perishable product, due to the short shelf lives and special storage requirements of these products. When placing replenishment orders for fresh produce, retailers aim to strike a balance between ordering too few product units, resulting in unsatisfied customers, and ordering too many product units, leading to expired products discarded as waste. The losses associated with food waste are a severe concern in the retail industry. In February 2016, France made news headlines as the first country to prohibit large supermarkets from discarding unsold expired food products [34]. French supermarkets are now compelled by law to donate these products to charities and food banks.

The complex periodic decisions that retailers are faced with in respect of the management of their product offering are interrelated and may be integrated. Deciding on the variety of products



FIGURE 1.1: Retailers rely on the colourful display of the fresh produce departments to welcome customers in a retail outlet [121].

and brands to stock in a store should be done before pricing decisions related to these products may be performed. In terms of inventory management, retailers are required to establish target inventory levels, replenishment order schedules and review periods. Closely associated decisions also have to be made in respect of the allocation of limited shelf space in a retail outlet among the products sold in the store. Optimisation and streamlining efforts are required in these interrelated decisions in order to maximise the profits of and minimise wastage by retailers.

1.2 Project aim and scope

The complexity of the aforementioned decisions in the fresh produce department of a retail outlet may be attributed to the short shelf lives of fresh produce, operational constraints and a limitation on the amount of shelf space available. Retailers may gain a competitive advantage through the efficient management of shelf space in their fresh produce departments.

A special type of retail outlet is considered in this thesis. This type of retail outlet has no backroom inventory and offers a limited product range in its fresh produce department. All product units that arrive at the store are immediately displayed on the shelves. The distinctive short shelf life of fresh produce should be taken into account, as well as lead time associated with product replenishment and product pack sizes. The aim in solving the problem of how much of each product to stock is to satisfy forecasted product demand over a decision period as closely as possible, while taking profit into account and minimising product waste, all with a fixed amount of shelf space available. A specially tailored, objective method is required to construct a replenishment order schedule for the fresh produce department of this type of retail outlet. The design and validation of such a method is the research aim of this thesis.

A real retail outlet within the aforementioned class of retail outlets fulfills the role of industry partner in a special case study considered in this thesis. Although the characteristics of this retail outlet influence the nature of the mathematical model proposed, the model is designed to be generic and is also applicable to other retail outlets.

The mathematical model is subject to several assumptions, which are described as part of the model formulation. The model is embedded in a computerised concept demonstrator of a *decision support system* (DSS) which takes forecasted demand data as input and recommends

a fresh produce replenishment order schedule as output. The concept demonstrator is applied to real demand data in a case study so as to demonstrate the practical workability of the DSS proposed and the quality of its replenishment order recommendations.

1.3 Thesis objectives

In order to achieve the project aim according to the methodology described in the previous section, seven objectives are pursued in this thesis, namely:

- I To *conduct* a literature review on the following relevant topics:
 - (a) The functioning of a retail outlet and the retail industry in general,
 - (b) The unique characteristics of fresh produce, which may influence replenishment order decisions,
 - (c) The complexities associated with retailer decisions pertaining to inventory management and shelf space allocation,
 - (d) Existing mathematical modelling approaches toward retail replenishment and decision support,
 - (e) Suitable solution techniques that may be employed to solve combinatorial optimisation problems,
 - (f) Techniques and approaches typically followed in the validation of mathematical models, and
 - (g) Development guidelines for DSSs.
- II To *determine* a set of suitable retailer objectives, considerations and constraints that should preferably be adhered to during fresh produce replenishment order decisions. Pursuit of this objective should be based on the review of the literature carried out in fulfilment of Objective I and discussions with a representative of the retail sector.
- III To *formulate* a general mathematical model for replenishment order schedules of fresh produce in retail outlets with limited or no backroom storage space based on the objectives, considerations and constraints identified in pursuit of Objective II.
- IV To *develop* an appropriate solution methodology for solving the mathematical model of Objective III and to *validate* the solution methodology by solving small hypothetical fresh produce replenishment order problem instances.
- V To *design* a DSS of which the working is based on the mathematical model of Objective III and the solution methodology of Objective IV, which yields as output suitable replenishment order schedule recommendations for fresh produce.
- VI To *apply* the DSS of Objective V to a special case study involving an actual retail outlet of a large local retailer in order to demonstrate the practical workability of the DSS.
- VII To *recommend* possible follow-up work based on the work presented in this thesis, which may be pursued in future.

1.4 Thesis organisation

A diagram of the organisation of material in this thesis is provided in Figure 1.2. Connecting lines between chapters in the figure indicate a dependency of the contents of a chapter on the contents of previous chapters.

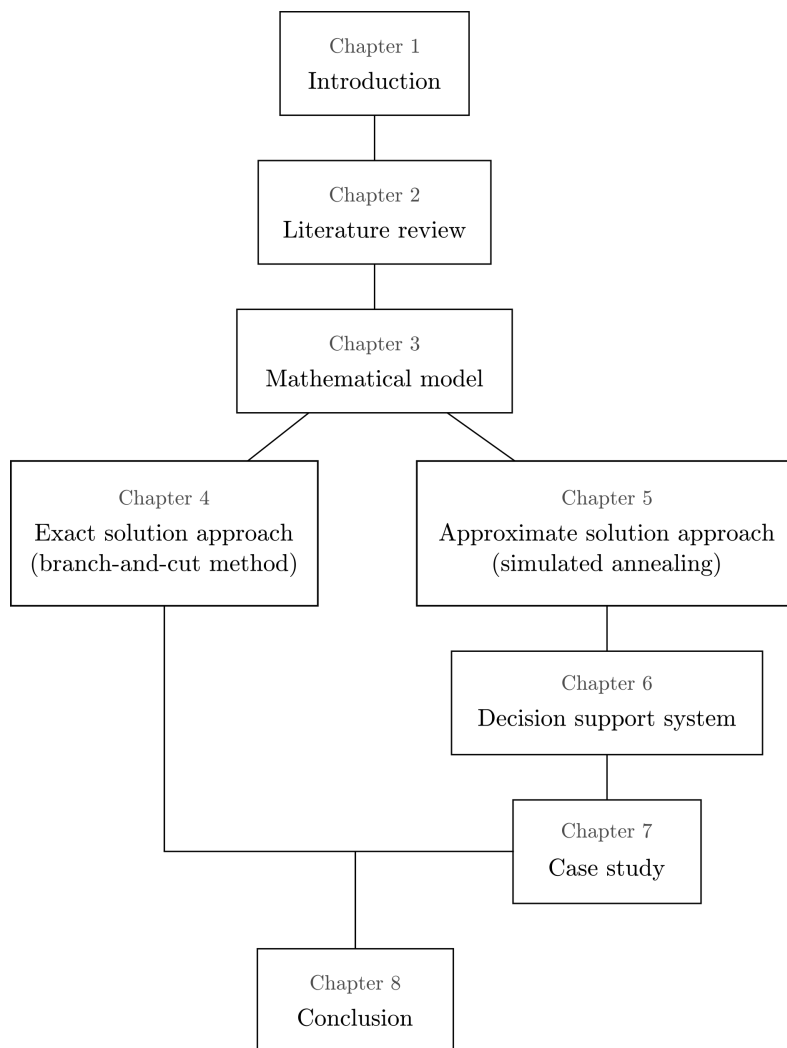


FIGURE 1.2: A diagram of the thesis organisation.

Chapter 2 contains a literature review on topics that are relevant to various aspects of the problem described in §1.1 and §1.2 as well as the thesis objectives outlined in §1.3. More specifically, the literature review is focussed on the retail industry, the management of a fresh produce department within a retail outlet, inventory management and retail shelf space allocation, exact and approximate optimisation model solution techniques, model validation techniques and design guidelines for DSSs. The retail industry is described in §2.1 with reference to basic retail definitions, general operations at a retail outlet, customer behaviour in a retail outlet and the nature of the South African retail sector. Fresh produce is typically seen as a perishable product (described in §2.2.1) and is susceptible to product expiry, which contributes to food waste (discussed in §2.2.2). There exist several fields of study in the literature that are related to the replenishment order decisions of retailers, namely product assortment, price planning, inventory management and shelf space allocation. These respective fields of study are reviewed briefly in §2.3. Solution techniques for combinatorial optimisation problems are briefly described in

§2.4, distinguishing between exact solution techniques and approximated solution techniques. Particular emphasis is placed in this section on the specific techniques that are actually applied in later chapters of this thesis. Mathematical model validation techniques are discussed in §2.5, while general considerations pertaining DSS design and development are discussed in §2.6.

The formulation of a novel mathematical model in support of a retailer's fresh produce replenishment order decisions is documented in Chapter 3. The formulation draws from the discussions in the previous two chapters, especially in respect of the problem characteristics and related mathematical models in the literature. Assumptions and practical considerations which influence the mathematical model are acknowledged in §3.1. The objective function of the model is based on three conflicting criteria typically considered by a retailer when deciding on fresh produce replenishment order schedules, as explained in §3.2. Shelf space is a limited and valuable resource in a retail outlet. This resource therefore features centrally in the model constraint derived in §3.3. A further set of model constraints measure the extent to which demand satisfaction is maintained, as described in §3.4. The conservation of stock is also modelled as a constraint set in §3.5. The modelling of product waste is finally explained by means of a small example in §3.6 before the entire mathematical model is presented as a chapter summary in §3.7.

Two primary solution approaches are followed to solve the mathematical model of Chapter 3, hence the branch in the diagram of Figure 1.2. The first of these approaches is an exact solution methodology implemented in IBM ILOG CPLEX Optimization Studio (CPLEX), which is described and applied in Chapter 4. The computer implementation of this solution approach is discussed in §4.1, while hypothetical product data for use in hypothetical problem instances are provided in §4.2. The numerical results of three problem instances returned by the exact solution methodology are analysed in §4.3. An approximation method and a problem-specific heuristic are presented in §4.4 for the model of Chapter 3 in an attempt to use CPLEX to solve the mathematical model approximately, before it is concluded in §4.5 that a more sophisticated approximate solution approach is required.

The second solution approach employed to solve the mathematical model of Chapter 3, namely the method of simulated annealing, is presented in Chapter 5. The development of the algorithm is documented in §5.1, dedicating subsections to each of the respective components of the simulated annealing algorithm, namely the constraint handling technique adopted, the method of initial solution construction, the simulated annealing move operator employed, the acceptance rule for neighbouring solutions adopted, the method of initial temperature selection, the cooling and reheating schedules incorporated and, finally, the search termination criteria enforced. A pseudocode description and details pertaining to the implementation of the algorithm in R are furthermore provided. In §5.2, nine small hypothetical problem instances of increasing complexity are solved by both CPLEX and the simulated annealing algorithm in order to validate the implementation of the simulated annealing algorithm as a model solution methodology instead of the exact solution approaches implemented in CPLEX, as described in Chapter 4. Finally, a larger hypothetical problem instance is solved in §5.3 as a demonstration of the algorithm's operation, before the chapter closes with a brief summary in §5.4.

Chapter 6 is devoted to the design of a DSS for fresh produce replenishment order scheduling. The work of the previous three chapters is brought together in the design process. The aim of the DSS is to assist high-level managers of retail outlets in deciding on replenishment order quantities and timings for fresh produce, as well as to facilitate the repeatability and reproducibility of the work presented in this thesis. The architecture of the DSS is proposed in §6.1. In order to demonstrate the functioning of the DSS, a walk-through description of a concept demonstrator of the system, together with accompanying screen shots of its *user interface* (UI), is provided in §6.2.

A real-life case study is presented in Chapter 7, involving 72 products of the ambient section of the fresh produce department of a retail outlet in Grassy Park, Cape Town. Background information on this retail outlet is provided in §7.1. A description of the data and the necessary data preparation follow in §7.2. The design of an algorithmic parameter evaluation experiment in respect of the particular problem instance is described in §7.3. The aim in the experiment is to determine a set of suitable values for selected parameters in the simulated annealing algorithm when solving the mathematical model of Chapter 3 in the context of the case study problem instance. Numerical results pertaining to the experiment are presented in §7.4. Upon solving the model by the simulated annealing algorithm, incorporating the parameter values uncovered in §7.4, a fresh produce replenishment order schedule is recommended for the Grassy Park retail outlet over a one-month decision period. In §7.5, this replenishment order schedule is compared to the schedule that was actually employed at the retail outlet over the same period, and this is followed by a detailed discussion on the differences between and similarities of the two schedules. In §7.6, the effectiveness of the approximate solution methodology embedded in the DSS of Chapter 6 is ascertained in the context of the case study.

The thesis conclusion in Chapter 8 consists of a summary of the work presented in this thesis in §8.1. Feedback from an industry representative in respect of the validity of the mathematical model, as well as the desirability and practicality of the replenishment order schedule recommended by the DSS of Chapter 6 in the context of the case study of Chapter 7 is also provided. This feedback is incorporated in an appraisal of the thesis contributions in §8.2. Recommendations for future work are finally made in §8.3.

CHAPTER 2

Literature review

Contents

2.1 The retail industry	7
2.2 Fresh produce	12
2.3 Inventory management and retail shelf space allocation	16
2.4 Solution techniques for combinatorial optimisation problems	28
2.5 Model validation techniques	33
2.6 Decision support systems	35
2.7 Chapter summary	36

This chapter is devoted to a review of topics in the literature that are relevant to the problem described briefly in the previous chapter. A variety of characteristic aspects of the retail industry potentially influence the problem under consideration. A summary of the retail industry, with reference to various key definitions, general operations and customer behaviour, is therefore provided in §2.1.

Fresh produce exhibit unique characteristics, such as short shelf lives and high demands, which are reflected in the mathematical modelling of inventories of these products. Furthermore, product waste is a serious consequence of these characteristics. Section 2.2 contains elaborations on these subjects.

The formulation of the mathematical model for fresh produce ordering proposed in the following chapter draws from the retail-related fields of inventory management, product assortment selection, price planning and shelf space allocation. The respective organisational approaches followed in these fields, as well as the interdependence among them, are described in §2.3. Exact and approximate solution approaches that are appropriate for solving the mathematical model proposed in this thesis are furthermore presented in §2.4.

Techniques that may be employed to validate the proposed mathematical model are described in §2.5. Finally, general requirements and essential considerations pertaining to the design and validation of a DSS are discussed in §2.6, before the chapter closes with a brief summary in §2.7.

2.1 The retail industry

The retail industry is one of the largest industries in the world [48] and is considered a highly competitive industry [68, 99]. This industry includes any business that sells goods or services

to consumers [75]. In South Africa, the retail trade industry refers to the selling of new goods and the reselling of used goods to the general public, for personal or household use [50, p. 3].

2.1.1 Basic retail definitions

One of the most commonly used acronyms in the retail industry is SKU, an abbreviation for *stock keeping unit*, which may be described as the smallest management unit in a retail store [12] and refers to a single product unit. Products may be categorised into *product families* based on certain characteristics, often according to product brand or according to variety or flavour [141]. Fruit juices may, for example, be categorised into sizes such as 250 ml, 1.5 l or 5 l, or according to flavour such as orange juice, apple juice or grape juice. A *product category* refers to a set of products that exhibit similar characteristics, such as a bread category, a tinned fish category or a toothpaste category, and may contain several brands and SKUs [10].

There are many types of retail outlets [105]. For example, *department stores* offer a wide range of products, including electronic appliances and clothes, while *discount stores* typically offer fewer products, but at lower prices. *Speciality stores*, on the other hand, sell selected products and focus on satisfying customers, while *warehouse stores* sell limited products in bulk. *Supermarkets* sell household products and food items, which are usually displayed in different departments. Supermarkets have also been described as retail outlets which are medium to large in size and sell predominantly food [17]. A *mall* may be seen as a collection of retail outlets [105]. Online retailers are currently also a popular form of retail outlet. The variety of physical and online retail outlets that have emerged over the years as alternative to the traditional grocery store has contributed to a transformation in the retail industry [48].

The number of *facings* allocated to a product in a retail outlet is the number of displayed product units that a customer directly sees, which naturally excludes units that are placed behind other units [31]. It may also be described as the number of slots that are on the front of a shelf in a retail outlet. A specific number of these slots are allocated to each product by the management of a retail store [12].

A *planogram* is a visual representation of a retail shelf depicting the number of facings allocated to a product as well as their location relative to those of other products on the shelf [12]. A planogram is a traditional shelf space management tool, and is considered important in respect of assisting retail managers in their decision-making [66]. Planograms are usually generated by means of computer and allow the decision maker to form an idea of what the shelf layout will physically look like before organising products on the shelf and to ascertain the effect of moving products around on the shelves [51].

A distinction may be made between *private label brands* and *national brands*. Private label brands refer to products that are owned and sold by a single retailer, while national brands refer to products that are distributed by independent suppliers to retail outlets nationwide. A retailer would introduce private label brands in an attempt to improve the store image and to achieve greater flexibility in the establishment of prices and promotions [119], as well as to increase its ability to satisfy a wide range of customer preferences and to provide competitive substitutes for national brands [4]. Drawbacks associated with offering private label brands include exposure of the retailer to additional business risk and lower profitability associated with private label brands which are sold at lower prices than their national brand counterparts, although the latter drawback may be an outdated practice [17].

Space elasticity is the relationship between changes in the amount of shelf space allocated to a product and the corresponding changes in the sales of a product [42]. The notion of space

elasticity may be partitioned into two types of measures, namely *direct elasticity* and *cross-elasticity* [33]. Direct elasticity may be measured by changing the amount of shelf space allocated to a product and noting the changes in the number of product units sold. Cross-elasticity, on the other hand, takes into account the co-variational effects between products, and especially the resulting changes in the sales of other products when shelf space allocation changes are made to one particular product [10].

2.1.2 Operations at a retail outlet

Thousands of products are typically stocked in a retail outlet [77]. The average number of SKUs in a retail outlet has increased exponentially over the years [48]. A retailer has to make several important decisions pertaining to its products before a planogram may be constructed, namely which products are to be included in the assortment selection, where products should be located in the store, and how much space should be allocated to each product [12].

Rules of thumb are usually adopted by retailers to establish desirable product inventory levels, which is inaccurate and influences the efficiency of the entire supply chain [143, p. 36]. Product inventory is generally stored in two locations in a retail outlet, namely on the retail shelves or in a backroom storage space [45]. In this way, the retail shelves may be used to display a wider variety of products and some units of large product orders may be stored temporarily in the backroom until shelf space becomes available for their display [10]. Disadvantages of incorporating backroom storage include increased costs as a result of continuous monitoring of products and increased operational complexity. Items ordered by retailers are, however, often delivered in case packs, which has a significant influence on product ordering operations since the size of replenishment orders are required to be multiples of the case packs [45]. Orders for such products are therefore placed less frequently, and more space is necessary to store the units.

Profit margins on the products sold by retailers are not very high [28]. It is therefore important for retailers to maintain a balance between the different costs associated with carrying products. Retailer success is furthermore not exclusively defined by profitability. It rather depends on the ability to find a good balance with respect to the number of products displayed to customers as well as the period and position of the product display, which should be in line with a changing environment attributed to changes in customer demand, the availability of new products and actions of competitors [59]. Challenges that the retail industry face include the management of the short shelf lives of grocery products, the complexity associated with keeping track of products efficiently and the difficulty of temperature control in the retail supply chain [48]. Other challenges that influence the management of products include demand uncertainty and highly demanding customers [143, p. 1].

Demand for a product depends on two very general factors, namely customers' personal preferences as a result of marketing or previous experiences, and the display of products in a retail outlet which, in turn, affects customers' preferences [138]. Product demand may also be influenced by external factors such as the weather, the day of the week and product prices [143, p. 78]. Some authors distinguish between two types of demand [21]. The first type of demand is as a result of customer preferences, product availability and the influence of in-store displays. The second type of demand arises when customers are willing to comprise and purchase an alternative product when their preferred product is not available. In an earlier paper, a third type of demand was also included, namely random demand as a result of customers who do not uphold preferences [5].

Product demand may differ at the various outlets of a retailer as a result of differences in the local economies, differences in customer cultures and demographics, as well as differences in store

formats [2]. Retailers are furthermore subjected to the challenging task of demand forecasting. Inaccurate forecasting may lead to overstocking which may, in turn, force retailers to decrease their product selling prices, or it may lead to stockouts and unsatisfied customers [143, p. 13].

Several authors have claimed that the amount of shelf space allocated to a product influences its sales [42, 70, 167]. Several interesting findings have emerged from a meta-analysis of shelf space elasticity in the literature, performed by Eisend [42]. He concluded that shelf space elasticity exists, but varies across product categories and is more prominent in categories that are seen as impulse buys, than in everyday products. Furthermore, it was found that the amount of shelf space allocated to a product has a more significant effect in larger stores, and depends on the characteristics of the products. Increasing the amount of shelf space has a greater influence than decreasing it and, although more shelf space generally increases sales, the rate of increase decreases for larger amounts of shelf space.

Contributing to the earliest shelf space allocation research, Brown and Tucker [24] reportedly partitioned products into three classes based on their responsiveness to changes in the amount of shelf space allocated to them. Products that are sold at a rate that is independent of how much shelf space is allocated to them are classified as *unresponsive products*, whereas products whose sales rates are somewhat dependent on shelf space allocation form part of the class of *general use products*. Finally, *occasional purchase products* are only sold when a large amount of shelf space is allocated them so as to increase their visibility.

Retail outlets are generally rich in technological advancements which may be used to the advantage of retailers. Scanners are, for example, used to collect sales data at the point-of-sale, and it is possible to share these data immediately with the retailers' suppliers [29]. These scanners typically collect large amounts of data, which may be used to improve decision making [143, p. 103]. Technology is typically also incorporated to keep track of individual product units, and more specifically of the level of freshness of perishable products, by developments such as *radio-frequency identification* (RFID) tags that can communicate with a reader [129].

Various software programs are also available to be used by retailers for decision support in respect of product category and shelf space decisions. These software programs are not considered optimisation tools, but often assist in shelf space allocation decisions through simulation [134]. Heuristics are often followed in such software to calculate solutions. These heuristics are usually governed by simple rules that are based on what has been done in the past. The software packages often require manual interaction and are mostly only used to generate planograms [67]. The simplicity of these software products and the easy implementation of the solutions thus provided does, however, contribute to their popularity [77].

2.1.3 Customer behaviour in a retail outlet

If a customer visits a supermarket and encounters a stockout when searching for a specific product and brand, (s)he can either leave the store to find the product elsewhere, decide to purchase the product during a next visit, forget about wanting the product, buy a similar product of the same brand, or choose to buy another product brand [47]. Customers *switch* to an alternative product when they change their purchasing decision as a result of expansion in the variety of products offered in a retail outlet, while they *substitute* one product with another when the product that they are actually looking for is not carried in a store [138]. A customer's willingness to substitute depends on product characteristics and availability [143, p. 58].

Smith and Agrawal [152] claimed that substitution influences both customer service and product demand, since the level of service depends on the satisfaction of customer demand. In some cases,

it makes sense to sell products that are not necessarily exactly what customers prefer, but are at the same time preferred products for substitution. In other cases, it makes more sense to focus on the most popular items in order to satisfy the customer demand in respect of these items. Sachs [143, p. 58] agreed by stating that retailers deliberately stock fewer products in order to reduce inventory costs, thereby shifting demand to other products and forcing customers to substitute.

Customer service, which contributes to the competitive advantage of a retail outlet, may be measured in several ways, involving the prices of products, the quality of products, whether products may be customised, the variety of products and whether customers receive personal attention [152]. The environment of a retail outlet may attract customers, persuade customers to purchase items and convince customers to return to the same store [26]. The colours and lighting in the store, the store staff and the music played in-store all contribute to a customer's experience in the store [53].

Kök *et al.* [89] claimed that a customer's choice of products is influenced by the customer's perception of general product variety instead of the actual product variety in a store. They explained that this perception may, in turn, be influenced by the shelf space allocation, the presence or absence of a favourite product and the assortment arrangement. They also pointed out that product variety is more important to some customers than to others.

Burns and Neisner [26] argued that customer satisfaction is the primary determinant of whether or not a customer will revisit a retail outlet. Customer satisfaction in retail outlets is often based on the difference between a customer's expectation of a visit to the store and the actual experience of the customer, which is also influenced by the emotional responses of the customer to physical attributes of the products and the retail outlet [26]. Furthermore, customers also form an association between a retailer and the products of its private label brand, which means that a low-quality product in the private label range of a retailer may have a negative influence on a customer's perception of the retailer [17].

2.1.4 The South African retail industry

As in many other developing countries, South Africa experienced considerable expansion of its retail industry during the past two decades, which may be attributed to factors such as trade liberation after 1994 and extensive recent urbanisation [128]. The increased development of housing in areas surrounding urban areas has resulted in the development of shopping centres shifting from inner-cities to the residential areas surrounding these cities [50].

In an annual report that identifies the largest retailers in the world (based on several factors, such as financial data and geographical region), the 2016 Global Powers of Retailing Report [118], five top-performing South African retailers were among the top 250 in the world — Steinhoff International Holdings Ltd, Shoprite Holdings Ltd, Pick n Pay Stores Ltd, The SPAR Group Ltd and Woolworths Holdings Ltd.

In a report by PricewaterhouseCoopers [130], in which past and forecasted trends and challenges in South African retail were analysed, one of the main findings was that the growing black middle class drives an increase in total retail sales. It was also found that the high South African unemployment rate and the associated income equality are among the most significant hindrances to retail growth. Furthermore, the rest of Africa is considered a promising opportunity for the expansion of South African retailers.

2.2 Fresh produce

In the context of this thesis, *fresh produce* refers to fruit and vegetables. Retailers rely on the fresh produce department of a retail outlet to welcome customers and to convince them of the high quality of all the products sold in the store, since the fresh produce department is often located near the entrance of a retail outlet and contributes to a customer's first impression of the store [103]. Retailers furthermore want to portray the image of selling the best quality products at the lowest prices [16]. High-quality products and the consistency of this quality are what makes a fresh produce department successful [16, 103]. Customers are sensitive to the quality of fresh produce. They base their perception of this quality on the visual appearance of the products, and will choose to purchase the fresher items on display [131, 144].

2.2.1 Perishable products

Product classification with respect to product shelf life is often observed when managing products in a store. In this respect, products are categorised as either *perishable* or *nonperishable*.

Hillier and Lieberman [62, p. 870] distinguish between perishable products and *stable products*. Any product that theoretically remains sellable forever, because there exists no deadline for disposing of the product, is classified as a stable product. Products that may only be carried in inventory for a limited period of time are considered perishable products. Their primary example of a perishable product is a daily newspaper, which may only be carried in inventory for one day before it 'expires.' Hillier and Lieberman elaborate further by providing a list of products that may be categorised as perishable. This list includes flowers, fresh food, Christmas trees, seasonal clothing and seasonal greeting cards.

Typical edible perishable products are meat, seafood, dairy products, eggs and fresh produce (fruit and vegetables). Perishable products exhibit certain characteristics which may influence shelf space allocation decisions. Compared to nonperishable products, the most prevalent characteristic of a perishable product is its short shelf life [161]. In order to slow down the process of their deterioration, perishable products require special storage conditions. Deterioration occurs quicker in higher temperatures [131], which is generally avoided by displaying products on refrigerated shelves. Frozen products are not categorised as perishable products, because freezing reduces the deterioration rate significantly [161].

The quality and variety of perishable products displayed contribute to a customer's perception of the store quality and may be the primary reason why a customer prefers a certain store above another [98]. Fresh perishable products also portray an efficient supply chain and a high demand for the specific products. Often, new perishable products are procured to replenish shelves before all the products on display have been sold. This poses a decision as to whether new and old stock should be displayed together, which is a decision unique to perishable products [98]. Some retailers separate old and new stock, preferring to sell the older products at a discounted price. Products can be sold separately in the same store in order to prevent products of poor quality from affecting fresh ones. Alternatively, older products can be sold from a different outlet of the same retailer in a lower income area [81]. Stores which do not separate old and new stock do so to minimise administration related to product prices and in order to minimise the amount of shelf space allocated to a specific product [98].

Other product characteristics that have been identified as contributing toward distinguishing between perishable and nonperishable products include average weekly sales, the coefficient of variation in weekly sales, delivery frequency, case pack size and minimum inventory [161]. Higher

weekly sales and less variation in average weekly sales are typically associated with perishable products. The minimum inventory level of perishable products is higher, deliveries of perishable products occur more often and perishable products' case packs typically contain fewer units.

The supply chain of perishable products is referred to as a *cold chain*, and consists of several processes followed to maintain special conditions from the time of harvest until products reach the end customer [80]. A well-managed, temperature-controlled supply chain normally leads to a competitive advantage [8]. Although several advanced technologies exist for improving temperature control, factors such as delays in deliveries are difficult to control and negatively influence product quality [3]. An inefficient cold chain leads to waste [80], which is an unnecessary expense that should be avoided as far as possible.

The thoroughly researched subfield of inventory theory related to perishable products (also referred to as deteriorating inventory) is of specific relevance to the topic in this thesis. While *perishment* and *deterioration* are used interchangeably, *obsolescence* does not form part of this subfield of study. Deterioration refers to products that lose their value as a result of spoilage, evaporation, decay, degradation, *etc.* [132], while obsolescence occurs when sudden technological developments or newly introduced products cause other products to lose their value over time [13].

A review of the literature on the mathematical modelling of inventory control of perishable products yielded six comprehensive overview papers of particular importance. Nahmias [117] reviewed the early literature (up to 1982) on ordering policies for perishable inventories, primarily for products with fixed shelf lives (but also in some cases products with random shelf lives). Raafat's review in 1991 [132] focused on models developed for exponential decay of perishables classified according to model characteristics such as demand type and whether shortages are allowed. Goyal and Giri summarised improvements and developments in the field by analysing and classifying mathematical models that appeared during the period between Raafat's review and 2001. They classified models according to the type of deterioration and the type of demand. Fundamentally different in perspective from all the other reviews, Li *et al.* [97] distinguished between models developed for deteriorating inventory in a single enterprise and those developed for an entire supply chain. They emphasised the fact that significantly more models form part of the first category, but that the second is becoming more popular among researchers. Karmakar and Choudhury [83] reviewed deteriorating models that appeared in the literature up to 2010, focusing only on models that account for shortages in the inventory system. The most recent review of deteriorating inventory control by Bakker *et al.* [13] appeared in 2012 and built on that of Goyal and Giri [55] by following the same classification method for models that appeared since 2001. They discussed model characteristics that have enjoyed thorough investigation and made recommendations as to which realistic inventory system characteristics require more research attention.

For mathematical modelling purposes, perishable products are generally classified into three categories, namely products that decay proportional to their quantity, products with fixed shelf lives, or products with random shelf lives [163]. The first category refers to perishable products which decay at a rate that is directly proportional to the amount of the product present (also known as exponential decay). Examples of this category are radioactive materials and chemicals [56]. Products with fixed shelf lives have predetermined expiry dates and an entire batch of products (of the same age) perishes at the same time [102]. Medicines and most food items form part of this category. Perishable products have random shelf lives when the time to spoilage is not known beforehand and differs from product to product [55]. Fresh produce generally forms part of this last category [55].

2.2.2 Food waste

There exists a particularly challenging trade-off for retailers in respect of perishable products, between ordering too many and too few product units. Too many units result in waste, while too few units lead to stockouts [143, p. 57]. This section is focused on a discussion of the first scenario.

Buzby *et al.* [27] pointed out that several definitions of food waste exist. Some authors claim that food waste is a component of food loss, where the latter term refers to a decrease in the quality or quantity of food, while the former refers specifically to the deliberate disposal of food that is either still fit for consumption or has expired, usually by the final consumer in the supply chain. Other authors define food waste as any food in liquid or solid state, cooked or uncooked, which is discarded, including excess food, as a result of processing, storing and food preparation. In this thesis, food waste refers to fresh produce product units that are discarded as waste because they have reached the end of their shelf lives while still on display in a retail outlet.

A study by the Food and Agriculture Organisation of the United Nations, an organisation established in 1945 to reduce food waste [125], calculated that approximately one third of the food produced for consumption is wasted [57]. Wasting food is considered a problem, especially when viewed within the context of the severity of hunger and malnutrition in the world [125]. Food waste is often disposed through landfilling, which is a problem in its own right, considering the decreasing availability of land fill sites [101]. Food waste also goes hand in hand with the wasting of resources that are unnecessarily utilised for products that are discarded [116]. Food waste occurs at all the different stages of the food supply chain — during harvesting, during transportation and distribution, in storage, when processing food, in retail outlets and after purchase by consumers [125]. The category of fruit and vegetables is the largest contributor to food waste [116].

More food is wasted in developed countries than in developing countries [57]. In developed countries, food is mostly lost during the consumption stage. Consumers in developed countries generally purchase more food than necessary in retail outlets, and discard food that is still edible. The lack of coordination in food supply chains in developed countries also contributes to food waste. In developing countries, food waste occurs primarily in the earlier stages of the supply chain. Role players in these countries often do not enjoy the advantage of advanced harvesting techniques, cooling and storage facilities or packaging systems that help prevent food waste in developing countries. In South Africa, for example, the total cost of food waste is estimated at R61.5 billion per year [116], as the sum of the costs of all edible food waste in different stages of the food supply chain.

A study was performed to determine the causes of food waste that occur between suppliers and retailers [110]. Interviews were conducted with managers of food departments of retail outlets in the United Kingdom and Spain, two developed countries. In this stage of the food supply chain, much value has been added to the products, which increases the cost of the waste. Furthermore, waste from this stage of the supply chain typically ends up in landfills. In this study it was found that the amount of waste associated with a specific product depends on the natural characteristics of the product (such as shelf life and fluctuations in demand) as well as market trends (such as the current popularity of fresh produce and consuming products out of season). Several causes were identified for fresh produce waste in retail outlets, including products that exceed their shelf lives, inaccurate forecasting of demand, quality of products below standard and insufficient available shelf space. An improvement in the cold chain as a result of investing in reliable equipment, together with shelf management aimed at achieving

TABLE 2.1: Common fresh produce products categorised as producing large amounts of ethylene and/or as ethylene-sensitive products [43].

Ethylene producer	Ethylene-sensitive
Apples	Apples
Apricots	Apricots
Avocados, ripe	Avocados, ripe
Kiwi, ripe	Avocados, unripe
Nectarines	Bananas, green
Peaches	Kiwi, ripe
Pears	Kiwi, unripe
Plums	Peaches
Prunes	Pears
Quinces	Persimmons
	Plums
	Potatoes
	Prunes
	Quinces
	Watermelon

good rotation of products, should decrease fresh produce waste. Nahman and De Lange [116] pointed out that the processing and packaging of fruit and vegetables lead to the majority of food waste in South Africa, followed closely by losses during the distribution of these products.

Buzby *et al.* [27] provided a comprehensive list of the causes of food waste in developed countries. These causes range from microbial damage, excessive or insufficient temperature and exposure to light, to overstocking by retailers for promotions, customer confusion about expiry dates and high customer expectations for product quality, as well as poor handling of products, damaged or inappropriate product packaging and inefficient coordination from suppliers.

It is a well-known fact that an unripe avocado should be placed in the vicinity of ripe bananas in order to allow it to ripen quickly. The reason for this phenomenon and its relevance to the fresh produce department of a retail outlet are described in the remainder of this section.

The ripening process of fresh produce does not cease after it has been harvested [131]. The ripening process is influenced by the environment and the biological characteristics of the produce. Ethylene is a gas naturally produced by plants, which promotes physiological processes such as seed development, fruit ripening, leaf shedding and senescence [151]. Ethylene is also responsible for several effects that are perceived as negative by consumers, including overripe, soft fruit (in the case of mangoes, apples and strawberries), bitter taste (in the case of carrots and lettuce), tough flesh (in the case of asparagus), sprouting (in the case of onions and potatoes) and loss or change of colour (in the case of broccoli and avocado) [108], as well as unpleasant odours, rotting and shrinkage [151].

Differences have been noticed among the ripening processes of various types of fruit, especially in the production of carbon dioxide and ethylene, which has led to the classification of fruit as either climacteric or non-climacteric [15]. Climacteric fruit produce higher levels of ethylene and carbon dioxide during ripening and continue to ripen after harvesting, while non-climacteric fruit need to ripen fully on the plant before being harvested [127]. Several recent studies have, however, shown that the distinction between climacteric and non-climacteric fruit is not entirely clear as several non-climacteric fruits ripen in the same way as climacteric fruits [127].

An alternative method of classifying fresh produce is according to their production of and reaction to ethylene. Some products produce large amounts of ethylene, such as apples and nectarines, while others are classified as ethylene-sensitive, such as unripe avocados and watermelons [43]. Ethylene-sensitive products deteriorate quickly when they come into contact with ethylene [156]. Common examples within both these categories are shown in Table 2.1. The majority of products that produce large amounts of ethylene are also sensitive to ethylene.

In a retail outlet it is necessary to control the ripening process of fresh produce in order to deliver ready-to-eat produce to customers while minimising product waste [151]. Ethylene can be artificially added during the storage period of fresh produce. A variety of chemicals are also often used to decrease the ethylene production in fruit and vegetables [108].

2.3 Inventory management and retail shelf space allocation

The development of a fresh produce replenishment order schedule, which is one of the aims in this thesis, draws from various retail-related fields. A retailer has to decide which products to sell in the fresh product departments of its outlets — a decision referred to as *product assortment planning*, as explained in §2.3.1. Another decision required by a retailer involves the pricing of items, which is briefly described in §2.3.2. The actual replenishment order schedule and the underlying reorder policy are derived from the field of *inventory management*, which is discussed in §2.3.3. The limited amount of shelf space available to display fresh produce within retail outlets relates to the field of *shelf space allocation*, which is summarised in §2.3.4. Many of these decisions are interdependent and have been integrated in proposed mathematical models. This section closes with a discussion in §2.3.5 of the various forms of integration within the aforementioned fields of study.

2.3.1 Product assortment

Deciding on the variety of brands to be included in the product assortment is one of the main considerations in the management of a retail store [138]. Product assortment decisions refer primarily to which products a retailer should stock on its shelves [134]. Hariga *et al.* [61] defined the *product assortment problem* as determining the variety of products that should be displayed on the shelves of a retail outlet in order to maximise value towards customers, in order to maximise the retail outlet's profit.

Questions that retailers consider as part of product assortment decisions include details about the variety of products, such as the flavours, colours and sizes, to stock in the store, as well as information about the respective levels of product inventory to maintain [142]. Retailers are required to decide on the number of product categories to include in the assortment, the number of products in each category, as well as the number of units of each product to carry in the store [106]. The uncertainty of customer demand and the occurrence of substitution complicate product assortment decisions [152]. Typical constraints in these decisions include the total cost of products that may be held in inventory and the amount of physical space available to display and store the entire assortment [106].

Assortment decisions are influenced by the selling prices and cost prices of products, the preferences of customers and customer behaviour in respect of substitution [64]. Miller *et al.* [112] agreed that the customer satisfaction with a retail outlet's assortment depends on the products included in the assortment, the changing and uncertain preferences of customers and their willingness to substitute. A retailer's choice of suppliers also has an influence on the total costs of

products carried in a store, which necessarily also influences assortment decisions [168].

Because there is such a large number of products available for retailers to choose from when creating their product assortments, especially in comparison to the typical size of an assortment, the number of possible combinations of assortments is almost endless [112]. Furthermore, new products are continuously being introduced into the market [46]. Inappropriate heuristics are often applied by retailers to select a product assortment. These techniques typically focus on a selected number of product characteristics, randomly replacing unpopular products with alternative products or only selecting the most popular products [112]. An alternative technique in selecting a product assortment involves identifying the products and brands preferred by customers in the target market, while also taking into account that customers desire flexibility in their choices and are prone to substitution [106]. Offering too many products in a store may lead to the phenomenon *product pollution*, in which case it is necessary to eliminate some of the existing products from the assortment [46].

An important aspect of assortment planning is to incorporate the willingness of customers to substitute [67]. Due to substitution, the demand for each of the products in an assortment depends on the other products included in the assortment [157]. Substitution should also be taken into account in demand forecasting, which is required for profit estimation and is, in turn, taken into account when selecting a product assortment, since retailers aim to maximise their profits [150]. Some retailers are capable of changing their product assortments during the course of the selling period, as more information on customer preferences becomes available [145].

There exist several trade-offs in product assortment decisions. Offering a wider variety of categories, for example, may limit the number of products that may be included in each category [106]. Moreover, a wider variety of categories and/or products leads to an increase in customer satisfaction, but also contributes to higher operational costs [168]. Miller *et al.* [112] pointed out that retailers are faced with a trade-off between assortment variety and choice overload. Another trade-off in product assortment decisions is customer perceptions and preference, which should be balanced with a retailer's supply chain constraints and external factors such as economic conditions and competitors' actions [106]. Retailers should furthermore find a balance between the fixed costs incurred per product when adding products to the assortment and the profits associated with expanding the assortment [152].

Offering a wide variety of products is advantageous from a marketing perspective, but it is associated with high inventory and transportation costs [152]. A large assortment ensures an increased probability that a customer will find the product that (s)he is looking for, which translates to sales for a retailer [157]. When more products are carried in a retail outlet, it allows customers to establish their product preferences [112]. A large variety gives customers more choices. Some authors have, however, argued that too many choices may frustrate customers because of the additional cognitive processing required and because it decreases a customer's ability to discern between the products in the assortment [22]. Furthermore, when a retail outlet carries a large assortment, the customer demand is spread over the large number of products, which leads to increased variability in the individual product demands [157]. This variability in demand, in turn, increases inventory costs [142].

The products and brands included in a retail outlet's product assortment furthermore affect a customer's preference towards a retail outlet [23]. From the retailer's perspective, poor product assortment decisions may lead to the lowering of selling prices as a result of overstocking or it may lead to lost sales if too few of the popular products are included in the assortment [150]. On the other hand, if customers experience the product assortment of a retail outlet positively, they become loyal customers, which generates profits for the retailer [106].

2.3.2 Price planning

Deciding on the price of a product is one of the most important decisions that retailers are faced with [18]. This is clear when taking into account that retailers are interested in the maximisation of their profits [95] and that a product's attractiveness may be seen as function of its selling price [164].

Traditional pricing methods used by retailers included merely adding a fixed percentage to the cost price of a product to determine the selling price, decreasing product prices by fixed percentages according to a fixed cycle and setting prices in accordance with competitors' prices [95]. More recently, the focus has shifted from the profitability of individual products to the profitability of an entire product category [58]. When considering deteriorating products, product pricing, together with inventory management, are of particular importance [104].

Retailers often sell grocery products at prices equal to the product cost prices, or, in some cases, at prices lower than the cost prices, which results in a loss per product [90]. This notion is known as *loss-leader pricing* and is a technique used to attract customers to a retail outlet in the hope that they will also purchase the other products in that store. Retailers often plan for markdowns, which refers to prices charged for leftover products at the end of a period, in order to clear these products [1, p. 273]. Temporary price reductions is a popular technique adopted by retailers to gain a short-term competitive advantage [90]. Unfortunately, customers often come to know a retailer's markdown strategy and wait for the selling prices to decrease before making a purchase, to the dismay of retailers [95].

Product pricing is especially important when retailers add to their assortments by introducing products that are improved versions of existing products [82]. A retailer may establish different prices for the same products in different retail outlets, because the outlets operate in and serve different geographical areas [95]. The effect of the price of one product on the sales of other products in the same retail outlet should be taken into account [58]. This does, however, add significantly to the complexity of the decisions. Substitution among products is a critical aspect to consider in pricing decisions, since this phenomenon is vital for accurate forecasting of demand in order to determine product prices that will maximise profits [82].

Product prices have an influence on in-store purchasing decisions of customers [115]. When choosing between different flavours or sizes of products, customers partly base their decisions on the respective prices [164]. The demand for a product either varies over time, or it increases in relation to price decreases [104]. Raz and Porteus [136] based their mathematical model for product pricing on the fact that the lowest variability in demand occurs at either very high or very low product prices. Products that are sold at middle range prices are usually in line with competing products and the demand for these products are more predictable.

Levy *et al.* [95] elaborated on seven factors which should be taken into account when determining optimal product prices. *Price sensitivity* refers to the effects of changes in the price of a product on the demand for the product, and may change from one season to the next. *Substitution effects* reflect the influence of the price of one product on the demand for competing, substitutable products. Furthermore, the dynamic effects of *promotions* cause a trend in sales over time. Product prices typically vary over customer segments, which is referred to as *segment-based pricing*. The prices of products in a particular category may influence the sales of products in a complementary category, a notion referred to as *cross-category effects*. The *costs* incurred by retailers to purchase products as well as the *discounts* offered to their customers should of course also be taken into account when establishing product prices. Finally, the prices of products sold by *competing retailers* may also influence local pricing decisions.

Retailers should bear in mind that customers are interested in receiving value for their money [82]. Retailers should furthermore consider pricing decisions in conjunction with shelf space allocation decisions [115]. Finally, because product prices have such a significant influence on product demand, these prices should also be taken into account in inventory management decisions [164].

2.3.3 Inventory management

Hillier and Lieberman [62, p. 828] defined *inventory* as stocks of goods that are held for future sale or use. In a manufacturing environment, inventory also sometimes refers to materials that are stored, materials waiting for processing or materials undergoing processing. The notion of *inventory management* is prominent in the literature, and appears abundantly in operations research text books and specialised scientific journals [6].

According to Winston [165, p. 846], the purpose of inventory management is to establish management rules for minimising costs incurred to maintain inventory and to satisfy customer demand. Andersson *et al.* [6] pointed out that there are typically two conflicting goals at stake in inventory management, namely the satisfaction of unpredictable demand and the fact that production (or ordering from suppliers) is more effective when it occurs on a large scale.

The popularity of inventory management as a research field may be attributed to the practicality of inventory problems and the integrability of inventory control with a variety of other fields of study, such as scheduling, facility location, transportation and retailing [61]. Scarf [146] added that inventory is also considered important from an economical perspective.

Manufacturers, wholesalers, retailers, and any other companies that handle physical products should manage their inventory effectively [62, p. 828]. A large inventory is undesirable as products in inventory may be seen as assets that do not yield any return, but cost an organisation money to store, while some products may spoil or become obsolete [146]. The cost of carrying inventory sometimes amounts to up to a quarter of the value of the inventory [62, p. 828]. A large inventory does, however, provide the security of nearly always having the required volumes of products on hand, which leads to improved customer service, and lower costs may be associated with fewer replenishments and larger replenishment orders [146].

Lead time is an important notion in inventory modelling, is defined as the amount of time that passes between the time when an order is placed to replenish inventory and the time when the order is taken into inventory, and is applicable in cases where products are purchased or produced [62, p. 833]. Hillier and Lieberman [62, p. 828] defined *demand* as the number of units of a product in inventory that will be required for some use, such as sales. When modelling inventory and demand, they distinguished between *deterministic* and *stochastic* inventory models. Deterministic inventory models are applicable in situations where product demand is known completely in advance of a decision period or may be forecasted accurately. Stochastic inventory models, on the other hand, are applicable to inventory systems where the product demand cannot be predicted, but is rather considered a random variable with a known probability distribution. Stochastic models are also sometimes referred to as probabilistic models [169]. Inventory models may further be classified according to the method employed to monitor inventory levels. *Continuous review* models refer to the case where the inventory level is continuously monitored and orders are placed as soon as the inventory level reaches a prespecified reorder point. If, however, the inventory level is reviewed at fixed intervals (typically weekly) and ordering decisions are only made after such reviews, the class of models is referred to as *periodic review* models [62, p. 833].

When demand for a product exceeds the available inventory, *backlogging* may occur. Backlogging is the notion of retaining a customer order for a product that is not currently available in stock until the inventory is replenished and the demand for the product may be satisfied [62, p. 832]. If backlogging is not allowed, the demand for unavailable products is described as lost sales, which are never satisfied, and the situation is referred to as a *shortage* [146], or a *stockout* [165, p. 847]. The occurrence of stockouts has been recognised as a major concern in the retail industry, especially since customers desire accessibility to a variety of products and store loyalty plays a significant role in customer behaviour. Retailers therefore spend their resources on their supply chains in an attempt to improve on-shelf availability of products [47]. Different types of costs may be incurred when a stockout occurs, namely costs reflecting the profit that is lost because customer demand cannot be met, the cost of rectifying the mistake by placing a special order, or the loss of future support by the affected customers [165, p. 868].

Variability and uncertainty in inventory systems may be attributed to factors such as the unpredictability of customer demand, potential product returns from customers, mistakes in orders delivered by suppliers and unforeseen increases in lead times, which depend on the capabilities of the suppliers [146]. *Safety stock* is additional inventory kept on hand to make up for unreliable demand information or short-term demand changes and variability in supplies [162]. Sachs [143, p. 13] added that safety stock may also be used to compensate for inaccurate forecasting of demand by retailers.

Hillier and Lieberman [62, p. 830] explained that six costs are typically included in inventory models that influence the profitability of inventory policies. *Ordering cost* is the total cost of purchasing or producing products. This cost may simply be proportional to the number of products ordered or manufactured, where a unit cost is associated with a product. The ordering cost may alternatively comprise two parts, namely a fixed setup cost and a product unit cost. In this case, the setup cost refers to the administrative cost of placing an order, or to the costs incurred to set up and initiate production. Secondly, *holding cost* (also sometimes referred to as *storage cost*) results directly from the storage of inventory. Elements that are often included in this type of cost are floor or shelf space cost, the cost of protection of stock, the cost of tied-up capital, insurance premiums and taxes. Holding cost may be calculated over the entire decision period or may be calculated continuously. The third cost, *shortage cost*, arises when the product demand exceeds the available inventory, and is also sometimes referred to as the *unsatisfied demand cost*. This cost is either calculated as the cost of backlogging an order, or the lost opportunity cost associated with unmet demand. Although the fourth cost is actually a negative cost and often not included in inventory models, *revenue* from sales may be considered a function of product prices and demands. The fifth cost, *salvage cost* or *salvage value*, refers to the cost or income associated with discarding a product at the end of a decision period. A cost might be incurred when disposing of a product, or leftover items might still yield an income when sold at some lower price. Finally, the *discount rate* is a cost that may be included to account specifically for the time value of the money that is tied up in inventory, which could alternatively have been invested somewhere to generate additional revenue.

The primary purpose of inventory modelling is to provide assistance in respect of two ordering decisions, namely *when* an order should be placed and what the *size* of the order should be [165, p. 846]. According to Silver [149], however, there are, in fact, three decisions that have to be made, namely how often the inventory should be reviewed, when a replenishment order should be placed and what the size of this order should be.

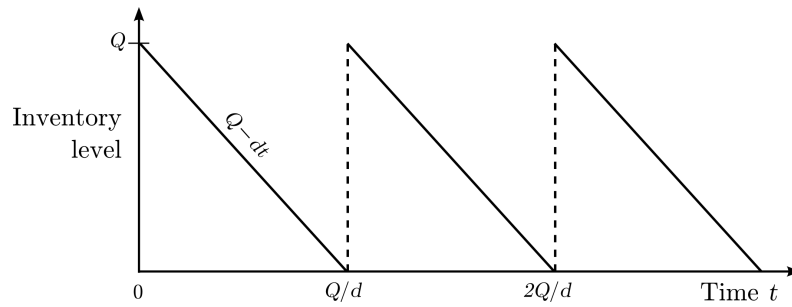


FIGURE 2.1: Inventory level of a basic deterministic continuous review inventory system.

Deterministic continuous-review inventory models

The most common inventory models in the literature are deterministic continuous-review models [62, p. 833]. Although deterministic analyses of inventory systems are often not satisfactory, deterministic models may nevertheless provide a foundation from which to build more complicated models that include the prevailing uncertainties [146]. The inventory level in the most basic form of such an inventory system over time is illustrated in Figure 2.1, where Q denotes the order quantity, d denotes the known constant demand rate for products held in inventory and t denotes time. The length of time between placing successive replenishment orders, called the *cycle length*, may subsequently be expressed by Q/d . Lead time and safety stock are ignored in the illustrated inventory system. The inventory starts at zero, and an order of size Q is placed at the beginning of the time period. From there on, an order is placed as soon as the inventory level reaches zero, so as to avoid stockouts. This process is repeated and the value of Q that minimises the overall cost of the inventory system may be determined by the well-known *economic order quantity* (EOQ) model [44], and is denoted by Q^* .

Several assumptions underlie the celebrated EOQ model [62, p. 833]. As mentioned, the model is applicable to deterministic continuous-review inventory situations only. Demand for product units occurs at a known constant rate, measured in units per unit time and denoted by d . When the inventory is depleted, an order of size Q is placed and all Q units arrive immediately. In the basic EOQ model, stockouts are not allowed. Lead time is assumed to be constant and may therefore be ignored in order to simplify calculations. A fundamental concept of the EOQ model is the fact that ordering occurs repetitively, which leads to the cycles that may be observed in Figure 2.1 [165, p. 850].

The costs taken into account in the basic EOQ model are ordering cost and holding cost. Ordering cost consists of an order setup cost, denoted by K , and a product unit cost, denoted by c , while holding cost is measured as the cost per product unit per unit time and is denoted by h . The aim in the EOQ model is to determine the order quantity, Q , in order to minimise the cost per unit time, T . The ordering cost per cycle may be expressed as $K + cQ$. Since the average inventory level during a cycle is calculated as $(Q + 0)/2 = Q/2$ and the cycle length is Q/d , the holding cost may be expressed as $h \times Q/d \times Q/2 = hQ^2/(2d)$. The total cost per cycle may therefore be expressed as $k + cQ + hQ^2/(2d)$, and subsequently the cost per unit time is

$$T = \frac{k + cQ + hQ^2/(2d)}{Q/d} = \frac{dK}{Q} + dc + \frac{hQ}{2}.$$

By setting the first derivative of T with respect to Q equal to zero, the value of Q that minimises the cost per unit time is found to be

$$Q^* = \sqrt{\frac{2dK}{h}},$$

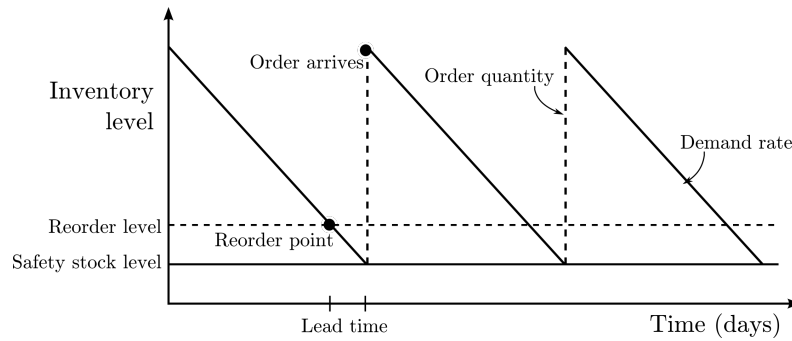


FIGURE 2.2: Inventory level of a deterministic continuous review inventory system where lead time and safety stock are incorporated.

which is known as the *EOQ formula*, proposed by Harris in 1913 [44]. Since it is often difficult to estimate the ordering cost and the holding cost, which influence the value of Q^* , it is reassuring to notice that small deviations from the actual EOQ value Q^* only result in a slight increase in the total cost [165, p. 853].

The basic EOQ model may be extended in several ways. Backorders may, for example, be allowed, since there may be situations where it makes sense from managerial and/or financial perspectives to plan short periods of stockouts [62, p. 836]. In such cases, the inventory level in Figure 2.1 will drop below zero when stockouts occur [146]. When lead time is taken into account, in order to prevent stockouts and to minimise holding cost, an order should be placed at a point which will ensure that the inventory level will be zero by the time the order arrives [165, p. 854]. This point corresponds to a *reorder point* which is associated with a *reorder level*. Furthermore, safety stock may be introduced to account for variability and uncertainty in the demand rate and the lead time. Figure 2.2 represents an extension to the inventory system of Figure 2.1 where lead time is taken into account and a fixed level of safety stock is maintained.

The basic EOQ model may also be extended to synchronise the ordering of multiple products. Roundy [140] proposed such a policy which stipulates that the reorder interval of a product (the time between reorder points) should be rounded to the nearest power of two. He proved that the implementation of this policy results in a replenishment order schedule that is 98% effective in satisfying customer demand on any data set, and leads to a decrease in overall cost due to improved coordination with respect to transportation of orders. Finally, ordering cost may include discount offered in proportion to the size of an order, which should be implemented accordingly in the EOQ formula [165, p. 859].

Deterministic periodic-review inventory models

In deterministic periodic-review models, the demand rate is not constant, and therefore the EOQ formula does not hold [62, p. 843]. Product demand per period is known, but is not consistent over all the periods considered. A deterministic periodic-review model presented by Winston [165, p. 969] takes into account that limited storage capacity is available for inventory. The approach of *dynamic programming* may be employed as a solution methodology for calculating order quantities that achieve the objective of minimising the total cost over the interval of periods considered [62, p. 843]. Dynamic programming is a technique used to decompose a large and complex temporal problem into a series of smaller decision problems and working backwards in time to obtain a solution [165, p. 961].

Stochastic continuous-review inventory models

Although stochastic inventory models are a better representation of the stochastic nature of demand, these models still only represent a limited part of real-life inventory systems. Such models are useful in providing guidance and insight to managers of inventory [146]. The EOQ model is, however, not applicable to an inventory system with stochastic demand, since the constant demand rate assumption is violated [165, p. 872].

In continuous-review models with stochastic demand, a so-called (R, Q) -policy stipulates the reorder point R and the reorder quantity Q [62, p. 866]. It is based on a traditional system where two bins are used to hold inventory of a specific product. One bin is used to satisfy the demand for the product, and as soon as it is empty, it triggers the placement of a replenishment order, while the second bin is used to satisfy the demand. This simple two-bin system has been replaced by computerised inventory systems where the inventory level is continuously monitored and the reorder point acts as the trigger to order the pre-specified reorder quantity. The EOQ model may be adapted to approximate this reorder quantity. The reorder point generally depends on a retail manager's objectives in respect of the level of service delivered to customers. This level of service may, in turn, be measured in several ways, including basing the measurement on the probability that a stockout will not occur during lead time, on the average number of stockouts during a period, on the average percentage of demand that is satisfied on time, or on the average delay in filling orders.

In some inventory situations, when the inventory level is close to the reorder level, it may happen that a large number of units is demanded at once, in which case the inventory level will drop far below the reorder level. In such cases, an (R, Q) -policy is insufficient, since stockouts will inevitably follow. An alternative policy is the so-called (s, S) -policy, which stipulates that an order should be placed whenever the inventory level is lower than or equal to a reorder level s , selecting an order quantity that is sufficient to increase the inventory level to S units [165, p. 896].

Stochastic periodic-review inventory models

Inventory models with stochastic demand are often applied to a single decision period only, corresponding to an inventory situation where a manager only has a single ordering decision to make [165, p. 880]. This type of stochastic model is often used to model the inventory of perishable products [146]. The model is also applicable to multiple periods, provided that the current inventory decisions are independent of future periods. Hillier and Lieberman [62, p. 870] presented such a model for perishable products. Only one product is considered in the model and, due to the perishable nature of the product, only a single time period is considered. Starting inventory is considered, and product units remaining at the end of the period may be salvaged for additional revenue. The demand is a random variable with a known or estimated probability distribution. The only decision variable is the order quantity. The objective in the model is to minimise the total expected cost, which includes ordering cost (setup cost and unit cost), holding cost (storage cost less salvage value) and shortage cost. The order quantity depends on the starting inventory and the probability distribution of the demand, and may be determined numerically. Winston [165, p. 881] described the use of *marginal analysis* to solve stochastic single-period models, which is a technique characterised by marginally increasing or decreasing the value of the decision variable and analysing the resulting effects in order to make a decision.

A policy widely adopted in stochastic periodic-review inventory situations (where more than one period is considered), is the so-called (R, S) -policy [165, p. 907]. This policy is based on the

on-order inventory level, which is the sum of the available inventory and the ordered inventory that is on its way (subject to some lead time). The on-order inventory level is also referred to as the *inventory position* [146]. Similar to the (R, Q) - and (s, S) -policies discussed above, R here represents the reorder point and if, upon review, it is found that the available inventory level is lower than S , an order is placed to bring the on-order inventory level up to S . Dynamic programming may also be employed to solve stochastic periodic-review models of this type [165, p. 1022].

The classical newsvendor problem also resides in this category of inventory models [86]. In this celebrated problem, a vendor selling newspapers has to make decisions regarding the daily order quantities in order to facilitate an appropriate trade-off between losing potential profit as a result of underordering and the potential cost of overordering, in a situation where the exact daily demands are unknown [62, p. 870].

More complicated inventory systems

When thousands of products have to be taken into account in the development of an inventory policy, products are sometimes classified according to their demands and contributions towards profits in order to reduce the size and complexity of the task [165, p. 911]. A retailer may also choose to manage the inventories of its respective outlets in one system. Challenges in such a system include the differences in the retail outlets' operations, their respective inventory policies and the allocation of limited inventory across the outlets [2].

Inventory system decisions may consist of multiple stages, referred to as *echelons*. For example, a first echelon may consist of the points where product units are manufactured or sold, a second echelon may consist of decisions related to the management of warehouses and a third echelon may consist of decisions in larger distribution centres [62, p. 848]. Careful coordination is required between the echelons of these so-called *multi-echelon inventory systems*, since it requires a bird's eye view of the inventory network.

An even broader view may be taken by considering how inventory arrives in the inventory system, which touches on the field of supply chain management. A *supply chain* is a network of companies that produce, manufacture, sell and deliver products and services to a specific customer segment, and consists of the suppliers, distributors, transporters, warehouses, retailers and the customers [126].

Complicated aspects of inventory management, such as interactions between products or intricate multi-echelon inventory systems, may be included in inventory-related decision considerations so as to better represent the actual systems, but might result in unmanageable models [62, p. 890]. Ziukov [126] pointed out that the inventory situation of each organisation is unique and that there is no standard model that may be applied to manage the features and limitations of all inventory situations.

2.3.4 Shelf space allocation

Shelf space is widely recognised as the most valuable resource of a retailer [115, 135]. It is often limited and the number of products to be displayed varies, which is why shelf space management is an important task for retailers [77]. The efficient allocation of a retailer's scarce resource of shelf space is a necessary advantage to a retailer in the competitive retail industry [91, 99]. When managed well, shelf space allocation attracts customers, prevents stockouts and increases a retailer's profits [77]. Furthermore, the majority of the customers of a retail outlet make their

final purchasing decisions in-store, a situation that retailers may use to their advantage when managing shelf space [66]. A vast amount of literature exists on the allocation of retail shelf space and the aim in this section is to provide a broad overview of what it entails and the different approaches followed to execute this important task.

The shelf space allocation problem

The shelf space allocation problem may broadly be defined as a decision-making problem in which the aim is to maximise a retailer's management objective subject to operational constraints [167]. More specifically, it refers to the allocation of a retail outlet's scarce shelf space among the products in a product category [36]. A third definition involves the number of facings allocated to each product stocked in a supermarket [135].

Shelf space allocation decisions may be made at two levels — first to allocate shelf space among product categories, and then to allocate shelf space to specific products within each category [49]. Shelf space allocation models often only pertain to decisions on the second level [21, 77, 159]. The first level then involves determining the product assortment [33]. Other models assume that the product assortment has already been selected by the retailer [115].

The shelf space allocation problem is also sometimes considered an extension of the well-known knapsack problem [167]. As in the knapsack problem, the shelf space allocation problem also deals with one primary resource which governs the planning, and the main difference between these two problems is that the shelf space allocation problem is also subject to policy constraints, in addition to the capacity constraints present in both problems [166].

Typical objectives in shelf space allocation models are to maximise profits, to reduce costs and to increase customer satisfaction [91], and sometimes multiple shelves are taken into account simultaneously [70]. Factors such as the demographics of the target market, seasonal influences on demand and strategic plans of the retailer also play a role in shelf space allocation decisions [79]. Many authors develop shelf space allocation models around the belief that product demand depends on the shelf space allocated to a product and other products related to the specific product, as well as the current level of inventory seen by customers [14].

Many of the shelf space allocation models in the literature take into account shelf space elasticity. As mentioned in §2.1.2, experimental studies have shown that the amount of shelf space or the number of facings allocated to a product has an influence on the product's sales [42]. Other authors have, however, argued that the location of a product on the shelf has a more significant influence on sales than how much shelf space the product occupies [91], a phenomenon referred to as *location elasticity*. Chen and Lin [33] critiqued the effectiveness of shelf space elasticities, emphasising the fact that the nonlinear nature of the relationship between shelf space and sales renders the mathematical models more complicated as well as the fact that many model parameters are required to incorporate space elasticities.

Cross-product elasticities are also sometimes considered in the mathematical modelling of shelf space allocation. This type of elasticity refers to the effect of the sales of one product on the sales of another product [79], and exists between products displayed on the same section of shelf space which are either complementary or serve as substitutes for each other [59]. Two products complement each other when an increase of sales in one product leads to an increase in sales of the other product [79]. On the other hand, two products may be substitutes, which implies that an increase of sales in one product leads to a decrease in sales of the other product.

Existing shelf space allocation models

Several mathematical models have been proposed since the 1960s to assist with the allocation of shelf space at retail outlets. These models vary in respect of a number of factors such as the way that demand is formulated, the problem constraints included, the space effects taken into account and the types of decisions considered.

In 1974, Anderson and Amato [5] incorporated a very detailed demand function into the shelf space allocation problem, and used it to determine both product assortment and shelf space allocation among a selection of products. In 1979, Hansen and Heinsbroek [60] proposed a similar model which determines an optimal selection of products, as well as an optimal allocation of shelf space, focussing on the fact that the number of product units to be displayed should be integers and by including a minimum number of displayed units for each product. One of the most cited models, proposed by Corstjens and Doyle [35] in 1981, takes into account a product's contribution towards profit, its responsiveness to changes in the amount of shelf space allocated to it (shelf space elasticity), cross-elasticities among products and costs associated with displaying the product. Constraints in this model include the total amount of shelf space available, and upper and lower display size limits per product. The aim of the model is to maximise overall profit as a result of product exposure to consumers. A literature review on the shelf space allocation problem revealed twelve models that are based specifically on the model developed by Corstjens and Doyle [1, 20, 25, 37, 70, 78, 99, 107, 159, 167, 166, 170]. Other models that follow different approaches are, however, also abundant in the literature.

In 1986, Zufryden [170] made use of dynamic programming and aimed to embed models for the product selection and shelf space allocation problems in a computer implementation, while taking demand and cost into account in the objective function. Borin *et al.* [21] maximised a retailer's return on inventory investment for each product category in their 1994 model and included the effect of stockouts, employing a heuristic based on the method of simulated annealing to obtain solutions. Urban [159] integrated product assortment and shelf space allocation with inventory theory in 1998, advocating that product demand depends on inventory displayed. In 1999, Yang and Chen [167] built on the model of Corstjens and Doyle [35], focussing on applicability in the retail industry by proposing an integer programming model and including a survey among store managers. Bookbinder and Zarour [20] followed a shelf space allocation approach that is fundamentally different to the models mentioned thus far — by considering the contribution of product units towards profit, known as the approach of Direct Product Profitability, and accounting for discounts, promotions, *etc.* In 2004, Irion *et al.* [78] also extended the model of Corstjens and Doyle [35] by taking into account the effect of displayed inventory on demand. Reyes and Frazier [137, 138] proposed shelf space allocation models in 2005 and 2007. The novelty of the first model is that it considers the impact of consumer behaviour demand [138], while the second one is a goal programming model that facilitates the tradeoff between customers service and profitability [137]. Also in 2005, Hwang *et al.* [70] proposed a model that combines shelf space allocation and inventory control, and they included the effect of a product's location on the shelf. The model proposed by Hariga *et al.* [61] in 2007 is an integration of a shelf space allocation model and an inventory model, with decision variables for determining product assortment, order quantities, backroom inventory and display locations.

Bai and Kendall [11] proposed a model in 2007 for the shelf space allocation of fresh produce by combining a deteriorating inventory model with a shelf space allocation model. Their assumptions differ from those underlying previous attempts at modelling shelf space allocation in the literature. In previous attempts, fresh produce had traditionally been considered a special class of perishable products, which implies a fixed deterioration rate and that all products have

the same value unless they have expired. Bai and Kendall [11] claimed that it is possible to predict the expiry dates of fresh produce, by using advanced cooling technology in the fresh produce supply chain. Their model is built on the assumption that the freshness condition of fresh produce decreases continuously although the value is not entirely lost by the time the products expire. They also assumed that the demand for fresh produce depends on the volume and the freshness of the products displayed. Freshness is a consumer measurement of quality. In other models it has typically been assumed that demand depends on inventory as a whole, but due to the scarcity of shelf space, only a fraction of inventory can usually be displayed. In the model of Bai and Kendall [11], an ordering policy is determined for fresh produce as well as the amount of shelf space to be allocated to each product. They employed the generalised reduced gradient method to solve instances of the model. In a later paper, Bai, Burke and Kendall [9] also used a metaheuristic and a hyperheuristic to solve instances of the same model approximately.

Silva *et al.* [91] considered two dimensions of shelf space in their 2009 model by differentiating between shelf levels and shelf parts, which lie vertically and horizontally, respectively. Several shelf space allocation models appeared in 2010. Murray *et al.* [115] proposed a model that simultaneously yields two-dimensional shelf space allocation with detail about product facings based on product prices. Gajjar and Adil [49] employed the method of piecewise linearisation to reformulate a variation of the model by Yang and Chen [167], which excludes the cross elasticity and location effects of products and assumes that all products are readily available. Russel and Urban [141] focussed specifically on exact locations of products on shelves, since some researchers believe that it has a significant effect on product sales. In 2014, Castelli and Vanneschi [31] proposed a hybrid genetic algorithm with variable neighbourhood search for solving a very basic form of the shelf space allocation problem in which the objective function is to maximise profit subject only to limited shelf lengths as well as lower and upper bounds on product facings. Geismar *et al.* [51] developed a model for two-dimensional shelf space allocation in 2015 which may also be used to find optimal advertising displays for webpage designs, and includes the effects of product shelf locations and determines the maximum number of facings per product from a product's potential profit.

Although numerous models have been proposed for both shelf space allocation and inventory control, none of them was developed specifically for fresh produce prior to the model of Bai and Kendall [11]. Only one other similar model appears in the literature. Piramuthu and Zhou [129] proposed an extension to the model of Bai and Kendall in 2013. They drew into question Bai and Kendall's assumption that all the displayed items of the same product are of the same quality. According to Piramuthu and Zhou, items are exposed to varying environmental conditions and therefore advanced technologies should be employed to track individual item deterioration. In contrast to the approach of Bai and Kendall, they modelled demand as a function of both item freshness and allocated shelf space. Furthermore, they computed the effective amount of shelf space allocated to items of a specific product, which is less than the actual allocated shelf space because of the influence of deteriorated items on the product demand. The rest of their model is similar to that of Bai and Kendall.

2.3.5 Integration of related literature

Retailer decisions associated with product assortment, price planning, inventory management and/or shelf space allocation are often considered simultaneously in the models proposed by authors in the respective fields of study.

Chen and Lin [33] pointed out that both product assortment and the allocation of shelf space are important factors which influence customer purchasing behaviour. Borin *et al.* [21] claimed

to have proposed the first model in which product assortment effects and shelf space decisions are considered jointly. When product assortment decisions and shelf space allocation decisions are merged, the result is referred to as *category management*, a field of study which has grown in popularity over the years [134]. Category management may also include decisions related to inventory and stockout management [30]. Category management is sometimes defined as the management of an entire product category as a single business unit [1, p. 79].

Campo and Gijsbrechts [30] stated that the aforementioned retail decision areas are integrated as a result of several interdependencies. Large product assortments, for example, complicate shelf space allocation decisions and limit product inventory levels. Ramaseshan *et al.* [134] pointed out that assortment decisions, shelf space allocation decisions, ordering decisions and assortment review periods all depend on the available shelf space and accurate demand forecasting, which emphasises the interdependency among these decisions. Moreover, it has been shown that customer behaviour towards stockouts are based on the make-up of a product assortment and the allocation of shelf space, while customer responses towards changes in a product assortment are influenced by the consistency of product availability and the management of retail shelf space [30]. Bai *et al.* [9] combined ordering decisions and shelf space allocation decisions in their model in order to create a high-quality automated system which may be used to maximise the overall profit of a retailer.

Eroglu *et al.* [45] pointed out that the amount of backroom inventory in a retail outlet is influenced by the product order quantities, the corresponding reorder points and the allocation of shelf space. They furthermore stated that, since product pack sizes are fixed by the suppliers and limit the order quantities, and shelf space allocation decisions are made periodically, the only variable in the control of the retailer to change the short term operations of a retail outlet so as to minimise costs, is the reorder point. Hariga *et al.* [61] partitioned research on inventory control into three streams, namely pure inventory systems where decisions are only related to an ordering schedule, extended inventory models where demand depends on the inventory level, and inventory problems integrated with assortment and shelf space allocation models. Many inventory control models take the effect of inventory on sales into account [160].

Cachon [28] described the retailing challenge of managing the interaction among transportation costs, inventory costs and shelf space costs. If, for example, a retailer would like to decrease its transportation costs by placing larger replenishment orders less frequently, more shelf space is required and inventory costs will also increase, or the retailer will have to compromise on customer service. Murray *et al.* [115] developed a model in which product category decisions are optimised in respect of product prices, display facing areas, display orientation and shelf space locations.

2.4 Solution techniques for combinatorial optimisation problems

Since the mathematical model for product replenishment order schedules based on available shelf space proposed in this thesis takes the form of a combinatorial optimisation model, a brief overview of solution techniques for combinatorial optimisation problems is provided in this section. A distinction is made between *exact* and *approximate* solution methods. Exact solution techniques may be used to obtain guaranteed optimal solutions to optimisation problems, while approximate solution techniques are used to obtain high-quality solutions to optimisation problems within a reasonable time frame, usually for practical purposes, without guaranteeing optimality [154, p. 18]. A breakdown is provided in Figure 2.3 of various solution methods for optimisation problems that belong to these two categories. Detailed descriptions of the partic-

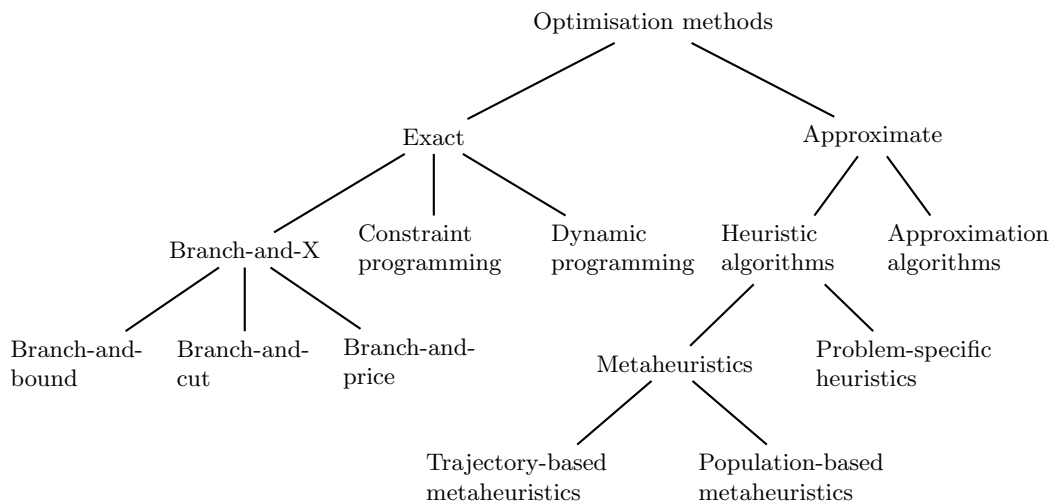


FIGURE 2.3: A classification structure of optimisation methods, adapted from figures by Dréo et al. [40, p. 19] and Talbi [154, p. 18].

ular instances within these two classes of solution approaches which are employed in this thesis to solve the aforementioned mathematical model are included in the overview discussion.

2.4.1 Exact solution techniques

In *dynamic programming*, a complex temporal problem is recursively partitioned into simpler subproblems and optimisation is performed on these subproblems in stages as a result of a sequence of partial decisions [154, p. 19]. As mentioned in §2.3.3, dynamic programming may be used to solve deterministic periodic-review inventory models. Dynamic programming is suitable for problems that exhibit the following characteristics [165, p. 967]: The problem may be divided into several temporal stages, requiring a decision at each stage, and a number of states is associated with each stage. Furthermore, the decision made during a stage in the problem describes the transition from the current state in the current stage into the next state in the next stage. The optimal decision in a current stage does not depend on historical decisions and states. Finally, a recursion function may be formulated for the problem in order to relate the cost or reward earned during an earlier stage to the cost or reward during all of the later stages.

A *constraint programming* model consists of a number of variables and a variety of constraints [62, p. 516]. Each variable may take on any value of a finite domain of values [154, p. 19]. There is considerable flexibility in respect of the types of constraints that may be included — from traditional mathematical constraints to logical and explicit constraints. Although constraint programming may be used to solve optimisation problems, it has traditionally been used to find feasible solutions to such problems instead of necessarily searching for optimal solutions.

The *branch-and-X* family of algorithms comprises branch-and-bound algorithms, branch-and-cut algorithms and branch-and-price algorithms as well as variations on these algorithms. Since the branch-and-cut algorithm is used to solve a combinatorial optimisation model put forward later in this thesis for determining replenishment order schedules for products in the fresh produce department of a retail outlet, this method is described briefly in the remainder of this section.

The branch-and-cut solution approach is a combination of the techniques of cutting planes and the branch-and-bound method [113]. The branch-and-bound method is a well-known technique

for solving integer problems, where branching refers to partitioning of the solution space of the original, large problem into smaller subproblems, and bounding refers to determining whether the optimal solution of one of these subproblems might be an optimal solution to the original problem [62, p. 492]. For *mixed integer programming* (MIP) problems, the subproblems are formed by sequentially relaxing the integer constraints in a sophisticated manner [113]. The branching process into subproblems gives rise to a tree, called the *search tree*, which grows by each iteration of creating new subproblems. The incumbent solution (the best feasible solution found up to the current iteration) is stored. Subproblems are dismissed from further branching when their optimal solutions are worse than the incumbent solution, or when no feasible solutions exist to these subproblems [62, p. 494]. A cutting plane is an additional constraint enforced over and above the constraints already present in the problem in such a manner that it reduces the feasible regions of subproblems without excluding feasible solutions to the original problem [62, p. 514].

2.4.2 Approximate solution techniques

As illustrated in Figure 2.3, the class of *approximate* solution techniques may be partitioned into the subclasses of *heuristic algorithms* and *approximation algorithms*. When applying approximation algorithms to an optimisation problem, a guarantee is obtained along with the final solution in respect of the closeness to optimality of the solution [154, p. 21]. A heuristic, on the other hand, is a method used to find relatively good solutions to large optimisation problem instances without any such guarantee and this class of techniques may be further partitioned into the subclasses of *metaheuristics* and *problem-specific heuristics*. The latter class contains solution methods that are tailored to solve a specific type of optimisation problem. In the class of metaheuristics, a distinction is made between *trajectory-based metaheuristics* and *population-based metaheuristics*. Hillier and Lieberman [62, p. 607] described a metaheuristic as a general solution method that controls a local search for improvement as well as a higher level search strategy aimed at ensuring that the solution space is effectively explored and that it is possible to escape from local optima. Popular trajectory-based metaheuristics include local searches [94], the method of simulated annealing [87], tabu search [54] and variable neighbourhood search [114]. Well-known algorithms in the category of population-based metaheuristics include genetic algorithms [63], particle swarm optimisation [85] and ant colony optimisation [38].

Since the method of simulated annealing is used to solve the combinatorial optimisation model put forward later in this thesis for determining replenishment order schedules for products in the fresh produce department of a retail outlet, this method is described thoroughly in the remainder of this section.

A vast amount of literature is available on the simulated annealing algorithm, which is based on the physical process of annealing metals or glass. Physical annealing starts by melting the metal or glass at a high temperature, after which the temperature of the system is gradually lowered until the metal or glass reaches a low-energy stable state and its physical properties are as desired [62, p. 628]. The strength of the atomic arrangement in the material depends on the cooling rate and when the initial temperature is too low or the temperature is lowered too quickly (referred to as *quenching*), the process results in a metastable state characterised by defects in the material structure [154, p. 126]. These defects may be removed by local reheating [40, p. 25]. As the system temperature is decreased, the energy level of the atoms in the glass or metal fluctuates but also decreases until a minimum energy state of these particles is finally reached [62, p. 628].

The simulated annealing algorithm was inspired by the resemblance between the complex struc-

ture of the solution space of a difficult optimisation problem and the material structure manipulated during the aforementioned process of physical annealing [40, p.23]. In the analogy between physical annealing and the simulated annealing algorithm, a solution obtained during the iterative search process of the method of simulated annealing corresponds to a state of the system during physical annealing, where the decision variable values are analogous to the positions of the atoms [154, p.126]. The energy level of the atoms at a point in time corresponds to the value of the objective function of the optimisation problem at hand during a specific iteration. Therefore, in the case of a minimisation problem, the mathematical aim of finding a feasible solution with a minimum objective function value is analogous to the physical aim of working towards a stable state where the energy level of the atoms in a material is as low as possible [62, p.628]. The stable or ground state is representative of the optimal solution, while a metastable state represents a local optimum [154, p.126] and is often described as the system being frozen [41]. Furthermore, the method of local search is analogous to quenching in physical annealing. In descriptions of the simulated annealing algorithm, references to the physical annealing process are often made. For example, in the simulated algorithm, one of the control parameters is referred to as the *temperature*, and the difference in *energy* between two solutions is iteratively observed for decision-making purposes.

The simulated annealing algorithm was proposed by Kirkpatrick, Gelatt and Vecchi [87] in 1983. Around the same time, however, a similar algorithm was independently proposed by Černý [32]. In both cases, the power of the algorithms was illustrated by solving the popular travelling salesman problem [111]. Kirkpatrick *et al.* explained how two specific techniques from the field of statistical mechanics, namely the Boltzmann distribution and the Metropolis algorithm, may be utilised in the simulated annealing algorithm [87]. The Boltzmann distribution governs the probability of a system being in a certain physical state at a specific temperature [40]. Furthermore, the Metropolis algorithm simulates the motion of atoms in a system's progression towards equilibrium at a specific temperature, by applying small random changes to an atom's position and probabilistically accepting it as a move [87].

The method of simulated annealing starts from an initial solution, either obtained from a greedy heuristic or generated randomly, and then continues to iteratively generate solutions in search of an optimal solution. The cost of the initial solution may be calculated as the value of the objective function in conjunction with optional penalties for possible constraint violations. During each iteration, a neighbouring solution is randomly selected by applying a move operator to the current solution. The cost of the candidate solution thus generated is compared to the cost of the current solution and the candidate solution is then either accepted or rejected. A candidate solution that yields an improvement in the cost function in respect of that of the current solution is always accepted, while a non-improving solution is accepted when a random observation from a uniform distribution between 0 and 1 is less than a probability that follows a Boltzmann distribution, which depends on the temperature of the system and the difference between the cost function values of the current solution and the candidate solution. For a minimisation problem, the acceptance probability is $p = \exp(\frac{Z_c - Z_n}{T})$, and for a maximisation problem it is $p = \exp(\frac{Z_n - Z_c}{T})$, where Z_c denotes the cost value of the current solution, Z_n is the cost value of the candidate solution and T the temperature parameter [62, p.627]. In a minimisation problem, the value of $Z_c - Z_n$ will be negative (similarly, for maximisation $Z_n - Z_c$ will always be negative) and as the value of T is decreased, the value of the exponent decreases which, in turn, decreases the acceptance probability over time, due to the nature of the natural exponential function. Therefore, at lower temperatures, the probability of accepting a non-improving solution is lower.

As in the physical annealing process, an appropriate temperature cooling schedule for the sim-

ulated annealing algorithm is essential [62, p.628]. The selection of the parameters in the temperature cooling schedule may be determined empirically in the quest to design an effective algorithm [62, p.628]. There exists a trade-off between the quality of solutions and the rate of cooling the temperature. Talbi [154, p.131] described several cooling schedules, including a linear cooling schedule [87], an adaptive cooling schedule [74] and a logarithmic cooling schedule [52]. The most popular cooling schedule, however, is the geometric schedule [154, p.131]. According to this schedule, the temperature T is decreased by applying the update rule $T \leftarrow \alpha T$, where α is referred to as the *cooling parameter* and typically takes on a value between 0.8 and 0.99 [41].

During the simulated annealing search process, random neighbouring solutions are iteratively selected and compared to the latest accepted solution until some stopping criterion is satisfied. The algorithm is memoryless, because information gathered during the search is not stored or used to make decisions [154, p.126]. For this reason, some sources mention that the different phases of the search may be modelled as a Markov chain — a new state depends only on the previous state [41, 158]. An alternative term for a Markov chain associated with a specific temperature is an *epoch* [155]. As in physical annealing, an initial temperature allows for many changes to the system and a gradual decrease in the system temperature steers the search in the direction of a locally optimal solution. Common stopping criteria include a limit on the number of iterations attempted, when a certain system temperature is reached and when the system has been reheated a predefined number of times.

A pseudocode description of the simulated annealing algorithm, adapted from Talbi [154, p.128], is provided in Algorithm 2.1.

Algorithm 2.1: Simulated annealing algorithm

Input : A cooling schedule $g(T)$ which is a function of the system temperature T , and the objective function f of a minimisation problem.

Output: A locally optimal solution to the minimisation problem.

```

1 Generate an initial current solution  $Z_c$ 
2 Set starting temperature  $T_0$ 
3 repeat
4     repeat
5         Generate a random neighbour  $Z_n$  of  $Z_c$ 
6          $\Delta E \leftarrow f(Z_c) - f(Z_n)$ 
7         if  $\Delta E \leq 0$  then
8              $Z_c \leftarrow Z_n$ 
9         else
10            Accept  $Z_c$  with a probability depending on  $\Delta E$  and  $T$ 
11    until New epoch criteria satisfied
12     $T \leftarrow g(T)$ 
13 until Stopping criteria satisfied
```

Simulated annealing is considered a valuable optimisation tool, because of its ease of implementation and its applicability to a wide variety of problems, often leading to very high-quality solutions [41]. The method is flexible in the sense that users may fairly effortlessly add or change model constraints after an initial implementation of the algorithm [40, p.44]. One of the major advantages of the algorithm is its ability to escape from local optima [40, p.23]. Solutions obtained by simulated annealing over a finite time horizon are, however, not necessarily optimal,

and the closeness to optimality of a solution produced by the method is never known. The various parameters necessary to implement the algorithm represent a disadvantage, as the onus lies with the analyst to adjust these parameter values so as to improve the efficiency and effectiveness of the algorithm [40, p. 44]. Another drawback of simulated annealing is the amount of computing time often required to obtain good solutions [40, p. 44].

Talbi [154, p. 48] summarised five categories of strategies for handling constraints in metaheuristics, which includes simulated annealing. According to *reject strategies*, infeasible solutions are immediately disregarded when found during the search. It makes sense to implement this category of strategies when infeasible solutions form a small part of the search space, since discarding infeasible solutions might otherwise result in closing off routes which may have led to better feasible solutions. The most popular strategies, however, are *penalising strategies*, which take feasible solutions into account albeit with an amount of penalisation. Two different penalty functions are often implemented within these strategies, taking into account either the number of constraints violated by an infeasible solution, or how far away it is from the feasible search space. *Repair strategies* consist of applying heuristics to transform infeasible solutions into feasible solutions. These heuristics are usually problem-specific and the effectiveness of these strategies depend on the nature of the problem. *Decoding strategies* seem more applicable to population-based metaheuristics, and make use of decoding functions to transform each representation of an infeasible solution into a feasible solution. Finally, *preserving strategies* are employed to ensure that only feasible solutions are generated and these strategies are therefore also problem-specific. Dréo *et al.* [40, p. 216] added some strategies to the previous five, namely strategies that convert constraints into objectives and solving problems as multi-objective optimisation problems as well as a method where the constraints are incrementally added and infeasible solutions are removed during the process.

2.5 Model validation techniques

The notion of model validation is present in several fields of study and is particularly popular in the field of simulation modelling. Although the model proposed in this thesis is a mathematical model and not a simulation model, simulation validation techniques and approaches are also applicable to the proposed model. In simulation modelling, *validation* is defined as the process of determining whether a simulation model accurately represents the real-world system that is studied, taking into account the particular objectives in the study [93]. *Mathematical model validation* has been defined as determining the extent to which a computer-implemented model accurately represents the real world from the perspective of the expected model applications [124].

Law [93] described several techniques that may be followed to develop valid and credible simulation models. Many of these techniques are also applicable to mathematical models. First, the problem should be formulated accurately. Subject-matter experts may be consulted throughout the development of the model in order to gather insight and relevant information. Furthermore, the underlying assumptions in the model should be documented. Finally, it is also advised that the output data of the model should be validated. Martis [109] also discussed several model validation techniques. The output of a model may be compared to the output of another model which is known to be valid. In the so-called *degenerate test* for validation, the removal of elements or resources from the model should be visible in the model output. *Face validity* refers to asking knowledgeable people about the reasonableness of the model output.

Landry *et al.* [92] proposed the idea of combining modelling and validation in a single process.

The authors distinguished between four stages in the modelling process, namely the problem situation, the conceptual model, the formal model and the solution. The *problem situation* is similar to the problem description in this thesis, and is merely a subjective definition of the real-world problem, which usually involves an opportunity for improvement of unsatisfying system behaviour. In the *conceptual model*, more detail is added to the problem description, such as the objectives of the model and which elements of the problem situation should be included and which should be excluded. The *formal model* is the mathematical or computer model developed from the conceptual model. The *solution* is obtained from the formal model and is considered the output of the modelling-validation process, from which recommendations may be made in respect of the problem situation.

Five types of validation form part of the modelling-validating process proposed by Landry *et al.* [92]. The main focus of *conceptual validation* is the assumptions made and theory applied when the conceptual model was developed. *Logical validation* is concerned with the accuracy of the formal model describing the conceptual model (and problem situation), which is often limited by the use of mathematics. *Experimental validation* is performed to determine the efficiency and quality of the solution techniques applied, as well as the type of solutions and the sensitivity of the solutions to changes in the model parameters. *Operational validation* reveals the usefulness of the solutions and recommendations obtained from the formal model. Finally, *data validation* deals with the availability, reliability and cost of the data that are used throughout the modelling-validation process.

When a metaheuristic, such as the simulated annealing algorithm, is used as an optimisation method, the user may experiment with a set of algorithmic parameters on the problem instances that (s)he is desiring to solve in order to select the best set of parameters [10, p. 71]. These types of experiments are referred to in the literature as the *configuration of parameter settings* [69], *parameter estimation* [167], *parameter optimisation* [147], *parameter identification* [124] and *parameter tuning* [88, 154]. In a later chapter of this thesis, the term *parameter evaluation* is preferred.

The parameter settings incorporated in a metaheuristic solution technique may have a significant influence on the effectiveness and the efficiency during the optimisation search process [154, p. 54]. It is often difficult to find the general best set of algorithmic parameters, since these parameters are usually very specific to a problem instance [10, p. 71]. The developers of computerised solvers typically focus on finding default parameter settings that are robust and acceptable across a variety of problem types [69]. Talbi [154, p. 54] distinguished between two strategies followed during parameter tuning. *Off-line parameter tuning* refers to the establishment of parameter values prior to the implementation of the metaheuristic approach. In *online parameter tuning*, on the other hand, the parameter values are changed dynamically or adaptively throughout the course of the metaheuristic execution.

It is theoretically possible to consider each combination of parameter values, but because of the computational complexity associated with such an exhaustive procedure, it is suggested that certain guidelines should be followed in order to choose parameter settings that enhance the performance of the algorithm under consideration [88]. Hutter *et al.* [69] performed such an experiment by using automated algorithm configuration procedures, where the parameter values of a target algorithm were iteratively varied and the algorithm applied to several problem instances, in order to measure and compare the performance of the algorithm under different parameter settings.

2.6 Decision support systems

A DSS may be defined as a computer technology solution used in support of complex decision making and problem solving [148]. A DSS may also be described as an information system developed to support and improve managerial decision making [7]. The three main components of a DSS are its *database*, its *user interface* (UI) and its *model base* [39].

Five types of databases may be distinguished [120]. A *flat file-based database* is the simplest type of database, and refers to human readable text formats and binary formats. Common examples of this type of database include *.csv* (comma separated values) files and spreadsheets. *Relational databases*, on the other hand, contain normalised data — the data normalisation is typically performed using SQL. A *hierarchical database* is suitable for data that may be organised in a nested format, such as a book with several chapters or a database of recipes. A *network database* is a more complicated version of the hierarchical database, and is less commonly used. Finally, the data in an *object-orientated database* consist of distinct objects with associated attributes.

A UI refers to all the channels of information that enable communication between a user and a computer [96]. The term *usability* is associated with UIs, and is defined as the measurement of the effectiveness of the system, the efficiency of resource utilisation and the user's level of satisfaction [122]. Lewis and Rieman [96] described a design process that may be followed in the construction of a UI. The first step of this process entails an analysis of the tasks to be performed as well as developing of an understanding and the nature of the user. Next, the primary tasks that are to be performed through the UI should be identified. Existing UIs may be used as a foundation from which a new UI is built. The design process further consists of a draft design of the UI, followed by a more detailed design which includes the decisions that the user will make. Next, a prototype may be built and iteratively tested and improved. Finally, the UI may be built, shared with the users and maintained.

The model base component may be partitioned into three stages involving model formulation, model solution, and an analysis [148]. During the *formulation* stage, the problem under consideration is translated into a format that is solvable by a computer algorithm. During the next stage, a suitable *solution* approach is found. And finally, the *analysis* stage refers to the presentation of the model in such a way that the problem and the solution are better understood. The optimisation techniques discussed in §2.4 typically form part of the model base component of a DSS.

Arnott and Pervan [7] differentiated between seven primary types of DSSs. A *personal DSS* is a small system developed for one or a few independent managers in support of a decision task. In a *group DSS*, technology allows a group of people to communicate and work effectively. A *negotiation DSS* facilitates opposing parties towards working together. When artificial intelligence is used to provide decision support, the DSS is referred to as an *intelligent DSS*. A *knowledge-based DSS* allows for the storage, retrieval, transfer and application of information in a group. *Data warehousing* refers to large-scale infrastructure used to provide decision support. Finally, *enterprise reporting and analysis systems* are used to provide executive information and other business reports.

Retailers require decision support for the efficient management of their products, their limited shelf space and the effects of customer decisions [67]. DSSs are also used for inventory management purposes [6]. Ramaseshan *et al.* [135] proposed a DSS for shelf space allocation and product assortment decisions, to which they refer as a *category management decision support tool*. In their DSS, required data are entered by the user by means of the UI. The user may select

one of two category management models, depending on the type of planning period adopted by the retailer. In addition to providing the model solution, the DSS also allows the user to generate reports reflecting the associated profits or the optimal review period, as well as a planogram.

2.7 Chapter summary

This chapter was devoted to a review of topics in the literature that are relevant to the problem considered in this thesis. In particular, the retail industry, fresh produce, the retail-related study fields of inventory management, product assortment selection, price planning and shelf space allocation, as well as exact and approximate optimisation techniques were discussed. Furthermore, techniques and considerations related to model validation and DSSs were also described.

Several concepts, such as the effect of shelf space management on customer behaviour, appeared in more than one section of the literature study, which highlights the interdependency of the different fields of study reviewed. Some of the notions discussed in this chapter are applied in the following chapters of this thesis, while other factors are merely seen as requirements to provide context, contributing to an understanding of the broader situation surrounding the specific decision area for which a decision support methodology is developed in this thesis.

CHAPTER 3

Mathematical model formulation

Contents

3.1 Model assumptions and considerations	38
3.2 The objective function	38
3.3 The shelf space constraint	39
3.4 Ensuring demand satisfaction	40
3.5 Modelling conservation of stock	40
3.6 Modelling waste	40
3.7 Chapter summary	42

The purpose of this chapter is to document the formulation of a mathematical model in support of inventory decisions by retailers in respect of their fresh fruit and vegetable departments. The objective function and constraints of the mathematical model are governed by the typical aims of retailers, the operation of retail outlets, the characteristics of fresh produce, and logic. The aim is to find a replenishment order schedule for the products in the fresh produce department of a retail outlet that is in accordance with the general aims of retailers, namely to pursue profitability, to satisfy customer demand and to minimise product waste.

Ideally, a replenishment order schedule should ensure that on every day of some decision period, the number of products on display equals to the forecasted demand for that day together with some volume of safety stock, in order to eliminate product waste and to allocate shelf space efficiently. In a real-world situation, however, this is usually not possible, because products are typically ordered in packs by retail outlets from a distribution centre, requiring that order quantities are multiples of these pack sizes. Moreover, order quantities should be kept low in cases where backroom storage space is limited in order to refrain from exceeding the fixed amount of shelf space available as well as to minimise the number of products that reach their expiry dates before they are sold.

General considerations and assumptions made in the formulation of the mathematical model are explained in §3.1. The formulation of the objective function is discussed in §3.2. The allocation of the limited resource, shelf space, is governed by a constraint, which is presented in §3.3. The constraint included to ensure demand satisfaction is discussed in §3.4. Constraints incorporated to ensure realistic flow of product units are derived in §3.5 and §3.6. The entire model is summarised as part of the chapter summary in §3.7.

3.1 Model assumptions and considerations

Let \mathcal{P} be the set of fresh produce products to be displayed at a retail outlet, and let \mathcal{J} be the set of days in the period over which inventory decisions should be taken. It is assumed that there is no backroom storage space available for fresh produce. It is envisaged that the length of the period will be of the order of one month, but may be shorter [153]. Several characteristics associated with each product are incorporated as model parameters, as is the total shelf space available. A discussion follows on the different aspects considered in the mathematical model.

It is assumed that all products are ordered from a distribution centre and take a known, fixed number of days to arrive at the retail outlet after having placed the order. These lead times should be taken into account when deciding upon a replenishment order schedule. It is also assumed that all products may be ordered and may be delivered on any day during the period. The lead time is considered a fixed product characteristic and the lead time of product $i \in \mathcal{P}$ is denoted by ℓ_i .

As mentioned in §2.3.3, the purpose of carrying safety stock is to deal with unreliable demand information and potential supplier delays. In the model proposed here, safety stock is assumed to be a fixed value associated with each product, which depends on the average demand for that product during the relevant decision period. There exist several ways to determine safety stock. The safety stock should be enough to provide for customer demand when orders do not arrive as planned, which is why formulas often used to determine safety stock involve multiplication of the average demand with the lead time. This usually results in a large safety stock value, and in the proposed model, the safety stock is therefore taken as half of this value. Instead of employing a separate variable to represent safety stock inventory, it forms part of the variable for product reserves in the model of this chapter. Furthermore, instead of aiming to have sufficient products on display for satisfying a specific day's forecasted demand, the aim is to have enough products on display to satisfy the daily demand *and* to have the safety stock on display. The safety stock required for product $i \in \mathcal{P}$ is denoted by s_i .

It is finally assumed that products are ordered from a distribution centre in multiples of some pack size, which is unique to each product. One pack of a particular product may, for example, contain fifty units of that product. This is accounted for in the model by taking the order quantity decision variable as the number of packs to be ordered. The order quantity is then multiplied by the product pack size in expressions where the number of product units is important. The order quantity decision variables are restricted to be integers, which is the reason for categorising the model as an MIP problem. The pack size of product $i \in \mathcal{P}$ is denoted by k_i .

3.2 The objective function

Retailers usually pursue three conflicting objectives in their fresh produce departments, namely to maximise the total profit gained from selling fresh produce, to satisfy forecasted demand and to minimise the amount of product waste in the fresh produce department [153]. By borrowing a technique from the realm of goal programming, these three objectives are translated into two goals within a single objective function in the model presented here. Goal programming is an extension of linear programming [139], and also resorts within the field of multi-objective optimisation [123]. Goal programming is applied when a decision maker has two or more contradicting objectives to consider, because it is often the case that the constraints representing these objectives limit the feasible region to such an extent that there does not exist a point where all the constraints are satisfied [165]. Therefore the constraints are rather interpreted as goals, and

since it is not possible to attain all the goals simultaneously, the sum of the deviations from the respective goal target values is minimised in the objective function.

The value of the proposed objective function may be interpreted as a lost opportunity cost, which should be minimised. Lost opportunity is measured in two ways in this case — by the extent to which the number of product units on display fails to meet demand (forecasted demand together with safety stock) and by the number of products discarded as waste. The difference between the sufficient number of product units that should be on display and the actual number of product units on display is called the *underachievement* of the products on display. In order to determine the cost of the lost opportunities over all the products over the entire decision period, the underachievement of a product is multiplied by its profit as this value represents the amount of income that the retailer could have received had the number of product units on display been sufficient, while the waste volume of a product is multiplied by the unit selling price to account for the price already paid for wasted products as well as the profit that could have been generated by the discarded products had they not been in excess of demand.

Let σ_{ij} denote the underachievement of the goal to have an ideal number of units of product $i \in \mathcal{P}$ on the shelf in order to satisfy demand on day $j \in \mathcal{J}$ and let w_{ij} denote the number of units of product $i \in \mathcal{P}$ discarded at the end of day $j \in \mathcal{J}$ as waste because of reaching their expiration dates. Furthermore, let p_i and a_i denote the unit selling price and the unit acquisition price of product $i \in \mathcal{P}$, respectively. The profit associated with selling one unit of product $i \in \mathcal{P}$ is then denoted by $p_i - a_i$. In mathematical terms, the objective is therefore to

$$\text{minimise } \sum_{i \in \mathcal{P}} \sum_{j \in \mathcal{J}} [(p_i - a_i)\sigma_{ij} + p_i w_{ij}]. \quad (3.1)$$

3.3 The shelf space constraint

In a typical retail outlet, a fixed amount of shelf space is usually allocated to the fresh produce department [98]. As a limited resource, shelf space may restrict the number of product units that can be displayed on any given day. Since there is no backroom storage space available for fresh fruit and vegetables, the number of displayed product units on a specific day is calculated as the sum of the starting inventory of that day and the order quantity arriving on the day. The total available shelf space is measured in squared centimetres and is a model parameter. As some products may be displayed in several layers, a stacking factor, representing the number of product units that may be stacked on top of each other, is another product characteristic.

Let r_{ij} denote the starting inventory of product $i \in \mathcal{P}$ at the beginning of day $j \in \mathcal{J}$. The above-mentioned two fixed product characteristics, namely the stacking factor and the unit display space required, are denoted by b_i and e_i for product $i \in \mathcal{P}$, respectively. The order quantity of product $i \in \mathcal{P}$ on day $j \in \mathcal{J}$ is denoted by q_{ij} — this is a model decision variable. In order to determine the order quantity arriving on day $j \in \mathcal{J}$, the quantity ordered the lead time number of days ago forms part of the stock availability constraint. Let S denote the total amount of shelf space available for fresh produce. It then follows that the shelf space availability constraint for day $j \in \mathcal{J}$ may be expressed as

$$\sum_{i \in \mathcal{P}} \left(\frac{r_{ij} + k_i q_{i,j-\ell_i}}{b_i} \right) e_i \leq S. \quad (3.2)$$

3.4 Ensuring demand satisfaction

The aim is to have enough product units on display every day to satisfy the demand of that day, as well as to have sufficient safety stock on hand. This goal may be achieved by the starting inventory, which is the inventory left over from the previous day, together with those product units arriving on the day in question. There is both the possibility of an underachievement and an overachievement associated with demand satisfaction. For every day and every product, the underachievement represents the number of units by which the number of displayed product units is smaller than the demand for that day, together with the safety stock. This allows for the use of inventory marked as safety stock to meet customer demand when necessary. The overachievement associated with demand satisfaction similarly represents the additional number of product units that are on display over and above the number demanded by customers, usually as a result of a large pack size or too much available shelf space. Only the underachievement is included explicitly in the objective function in (3.1). The overachievement is, however, included implicitly in the term for waste, as too many additional product units will result in product waste.

Denote the overachievement associated with demand satisfaction of product $i \in \mathcal{P}$ on day $j \in \mathcal{J}$ by μ_{ij} . Furthermore, denote forecasted demand for product $i \in \mathcal{P}$ on day $j \in \mathcal{J}$ by d_{ij} . Then the demand satisfaction constraint for product $i \in \mathcal{P}$ on day $j \in \mathcal{J}$ is expressed by

$$r_{ij} + k_i q_{i,j-\ell_i} + \sigma_{ij} - \mu_{ij} = d_{ij} + s_i. \quad (3.3)$$

3.5 Modelling conservation of stock

The inventory on display at the start of a new day is the ending inventory on display of the previous day. The ending inventory of the previous day is, in turn, equal to the starting inventory of the previous day together with the order quantity arriving on that day, less the units sold (the forecasted demand) during the day, less the product units discarded as waste at the end of the day. The conservation of stock may therefore be expressed by

$$r_{ij} = r_{i,j-1} + k_i q_{i,j-1-\ell_i} - d_{i,j-1} - w_{i,j-1}. \quad (3.4)$$

When calculated according to the expression in (3.4), however, the inventory at the end of a day may be negative. This may happen when the forecasted demand is larger than the actual number of product units to be displayed on that day, resulting in unsatisfied forecasted customer demand. The starting inventory of the next day should then be zero instead of a negative value, which is why the value of r_{ij} in (3.4) is adjusted to

$$r_{ij} = \max\{0, r_{i,j-1} + k_i q_{i,j-1-\ell_i} - d_{i,j-1} - w_{i,j-1}\} \quad (3.5)$$

for all $i \in \mathcal{P}$ and $j \in \mathcal{J}$. This adjustment allows for the calculation of the starting inventory as the difference between the product units that were on display during the previous day and the product units that were sold or discarded during the previous day.

3.6 Modelling waste

Since the goal is to have as little waste as possible, the value of the overachievement associated with product waste is equal to the value of product waste, which should be minimised. The

variables representing waste are therefore included in the objective function in (3.1) instead of incorporating separate variables representing the overachievement associated with product waste. Furthermore, underachievement is not possible in this goal, because that would imply negative values for product waste.

As mentioned above, waste here refers to units of a product that have been discarded because they have reached their expiry dates. This only occurs when the number of units ordered exceeds the sum of the forecasted demands for these days during which the product is on display, which is typical of products with exceptionally large pack sizes. For example, a product with a shelf life of three days, a forecasted daily demand of between twenty and twenty two units, and which is delivered in pack sizes of ninety units will generate a considerable amount of waste at the end of its three-day shelf life. In order to calculate the waste of a specific product at the end of a specific day, the number of product units that arrived a period of days ago equivalent to the length of its shelf life, as well as the starting inventory on the arrival day and the forecasted demand and waste during the entire period are required data. It is assumed that the oldest products are sold first. The calculation of waste referred to above is illustrated by a numerical example in order to elucidate the manner in which product waste is incorporated into the model.

Table 3.1 contains a comparison of four hypothetical products over a period of five days. For comparative purposes, the demands for all four products are assumed to be the same. Product 1 has a shelf life of three days and is delivered in pack sizes of ninety units. Product 2 comes in a significantly smaller pack size with the same shelf life as Product 1, while Product 3 comes in the same pack size as Product 1, but has a longer shelf life. Product 4 is included to illustrate the method of calculating waste. The table is populated by assuming that the starting inventory of all the products at the beginning of Day 1 is two units and that the first orders arrive at the start of Day 1, in the pack size of each product. From there on, the r_{ij} -values are determined by subtracting the forecasted demand (which is the number of units sold) and the waste of the previous day from the order quantity of the previous day. The q_{ij} -values have been chosen to reflect the situation where more product units are ordered when the value of r_{ij} is insufficient to satisfy the demand, except in the case of Product 4. The waste columns only contain positive values when units remain unsold for the duration of their shelf life and should therefore be discarded.

TABLE 3.1: *Product characteristics and daily values for waste calculation example. The values of k_i and t_i refer to the two characteristics of pack size and shelf life of product i , respectively, where $i = 1, \dots, 4$. The forecasted demand for product i on day j is denoted by d_{ij} , where $j = 1, \dots, 5$. Furthermore, r_{ij} is the starting inventory of product i at the beginning of day j , q_{ij} denotes the order quantity of product i arriving at the beginning of day j (without taking lead time into account) and w_{ij} is the waste of product i discarded at the end of day j .*

	Day 1				Day 2				Day 3				Day 4				Day 5			
	r_{ij}	q_{ij}	d_{ij}	w_{ij}	r_{ij}	q_{ij}	d_{ij}	w_{ij}	r_{ij}	q_{ij}	d_{ij}	w_{ij}	r_{ij}	q_{ij}	d_{ij}	w_{ij}	r_{ij}	q_{ij}	d_{ij}	w_{ij}
Product 1 $k_i = 90$ units $t_i = 3$ days	2	90	21	0	71	0	19	0	52	0	20	32	0	90	22	0	68	0	21	0
Product 2 $k_i = 30$ units $t_i = 3$ days	2	30	21	0	11	30	19	0	22	0	20	0	2	30	22	0	10	30	21	0
Product 3 $k_i = 90$ units $t_i = 5$ days	2	90	21	0	71	0	19	0	52	0	20	0	32	0	22	0	10	90	21	0
Product 4 $k_i = 90$ units $t_i = 3$ days	2	90	21	0	71	90	19	0	142	90	20	32	180	0	22	68	90	0	21	69

When excluding Product 4, it may be seen in Table 3.1 that the only waste is generated by Product 1 on Day 3, which can clearly be attributed to the combination of a large pack size and a short shelf life relative to the forecasted demand levels. The waste for Product 1 at the end of Day 3 is calculated by subtracting the demands for the last three days from the starting inventory of three days ago together with the order quantity that arrived three days ago, namely $2 + 90 - (21 + 19 + 20) = 32$. For Product 2, however, a similar calculation yields a negative number, as Product 2 clearly generates no waste, that is, $2 + 30 - (21 + 19 + 20) = -28$. This negative value should be adjusted to zero.

Suppose that, for some reason, a replenishment order schedule determines that ninety units of Product 4 should be delivered on each of the first three days of the five-day period considered in Table 3.1. Because of the demand for all five days being so low, the combination of the starting inventory and the order quantity arriving on Day 1 is more than enough to satisfy the demand during the first three days. The remaining product units that arrive on Day 1, together with the majority of the other product units (unnecessarily) arriving during the next few days, will be discarded as waste. This is not a realistic occurrence, but might exist in the model output. In that case, products already discarded should be taken into account when calculating waste. This is achieved by subtracting all the previous waste values over the time window representing the shelf life of the product.

The units of Product 4 arriving at the beginning of Day 1 expire at the end of Day 3. The value of the waste at the end of Day 3 may therefore be calculated as $2 + 90 - (21 + 19 + 20) = 32$ units. The units arriving at the beginning of Day 2 are not used to satisfy the demand of Day 2 or Day 3, because the units arriving on Day 1 are sufficient for this purpose. The units arriving at the beginning of Day 2 are only used to satisfy the demand of Day 4 (22 units), after which the remainder of these units ($90 - 22 = 68$) is discarded as waste. Similarly, those units arriving at the beginning of Day 3 are only used to satisfy the demand of Day 5, and because the units expire at the end of Day 5, the remainder ($90 - 21 = 69$ units) is discarded as waste. Taking into account that waste should not be a negative value, the general form of the constraint governing waste may therefore be expressed as

$$w_{ij} = \max \left\{ 0, r_{i,j-t_i+1} + k_i q_{i,j-t_i+1-\ell_i} - \sum_{j-t_i+1}^j d_{ij} - \sum_{j-t_i+1}^{j-1} w_{ij} \right\}. \quad (3.6)$$

The value of the waste of Product 4 at the end of Day 4, calculated according to (3.6), is $\max\{0, 71 + 90 - (19 + 20 + 22) - (32)\} = 68$, which corresponds with the value previously explained and validates the constraint.

3.7 Chapter summary

A mathematical model for determining a replenishment order schedule for the fresh produce department of a retail outlet was derived in this chapter. The different retailing aspects accounted for in the model were explained and motivated in detail. In summary, the model parameters and variables, together with their respective meanings, are summarised in Table 3.2, and the mathematical model is presented in its entirety. The objective in the model is to

$$\text{minimise } \sum_{i \in \mathcal{P}} \sum_{j \in \mathcal{J}} [(p_i - a_i)\sigma_{ij} + p_i w_{ij}] \quad (3.7)$$

subject to the constraints

$$\sum_{i \in \mathcal{P}} \left(\frac{r_{ij} + k_i q_{i,j-l_i}}{b_i} \right) e_i \leq S, \quad j \in \mathcal{J}, \quad (3.8)$$

$$r_{ij} + k_i q_{i,j-l_i} + \sigma_{ij} - \mu_{ij} = d_{ij} + s_i, \quad i \in \mathcal{P}, j \in \mathcal{J}, \quad (3.9)$$

$$\max\{0, r_{i,j-1} + k_i q_{i,j-1-l_i} - d_{i,j-1} - w_{i,j-1}\} = r_{ij}, \quad i \in \mathcal{P}, j \in \mathcal{J}, \quad (3.10)$$

$$\max\{0, r_{i,j-t_i+1} + k_i q_{i,j-t_i+1-l_i} - \sum_{j-t_i+1}^j d_{ij} - \sum_{j-t_i+1}^{j-1} w_{ij}\} = w_{ij}, \quad i \in \mathcal{P}, j \in \mathcal{J}, \quad (3.11)$$

$$\sigma_{ij}, \mu_{ij}, r_{ij}, q_{ij}, w_{ij} \geq 0, \quad i \in \mathcal{P}, j \in \mathcal{J}, \quad (3.12)$$

$$q_{ij} \in \mathbb{N}, \quad i \in \mathcal{P}, j \in \mathcal{J}. \quad (3.13)$$

Constraint set (3.8) ensures that the total space allocated to product units on any day during the decision period in question does not exceed the amount of shelf space available, while constraint set (3.9) represents the goal of satisfying demand each day. Constraints sets (3.10) and (3.11) represent the calculation of the waste and starting inventory values. Constraint sets (3.12) and (3.13) finally ensure that all decision variables are non-negative and that the order quantities are integer values.

TABLE 3.2: Summary of the model indices, parameters and variables.

Indices	
Symbol	Meaning
i	Index used to reference fresh produce products
j	Index used to reference days in the decision period
Parameters	
Symbol	Meaning
a_i	Unit acquisition cost of product i
b_i	Stacking factor of product i
d_{ij}	Forecasted demand for product i on day j
e_i	Unit display space required by product i
k_i	Pack size of product i
l_i	Lead time of product i
p_i	Unit selling price of product i
S	Total amount of shelf space available in the fresh produce department
s_i	Safety stock required for product i
t_i	Shelf life of product i
Variables	
Symbol	Meaning
q_{ij}	Order quantity of product i on day j
r_{ij}	Starting inventory of product i on day j
w_{ij}	Waste of product i discarded at the end of day j
μ_{ij}	Overachievement associated with demand satisfaction of product i on day j
σ_{ij}	Underachievement associated with demand satisfaction of product i on day j

CHAPTER 4

Model solution by CPLEX

Contents

4.1 Solving the mathematical model exactly	45
4.2 Product data for hypothetical test instances	46
4.3 Numerical results	48
4.4 Solving the mathematical model approximately	58
4.5 Chapter summary	63

The primary aim in this chapter is to review and apply an appropriate exact solution methodology to the mathematical model derived in Chapter 3. The off-the-shelf commercial solver IBM ILOG CPLEX Optimization Studio (CPLEX) is used to solve the MIP model exactly. A discussion of the input data and implementation details required to solve the model by CPLEX is included in §4.1. Hypothetical problem instance data involving ten distinct products are presented in §4.2, after which a sequence of small, increasingly more complex problem instances, created from these hypothetical data, are solved and the quality of these solutions are evaluated in §4.3. It is found that the exact branch-and-cut solution methodology employed by CPLEX is very costly in terms of solution time complexity. Finally, two approximate solution approaches are proposed and applied in §4.4 to the same instances as in §4.3 in an attempt to alleviate, to some extent, the computational complexity of the exact solution approach. The suitability of these heuristic solution approaches are then evaluated as alternative solution procedures.

4.1 Solving the mathematical model exactly

The mathematical model of Chapter 3 may be implemented in the *integrated development environment* (IDE) of CPLEX in order to obtain an exact solution. CPLEX is the preferred software platform in this respect, since it conveniently has its own IDE and may be downloaded for free under an academic license. It is also considered one of the industry standards in mathematical programming solution software [62, p 6]. CPLEX was developed by Robert Bixby to solve *linear programming* (LP) and *integer programming* (IP) problems [19]. The first commercial version of the optimiser was released in 1988. The simplex algorithm was implemented in the C programming language (leading to the name CPLEX), and the software was acquired by ILOG in 1997 and again by IBM in 2009; hence the long name IBM ILOG CPLEX [100]. Today, after many improvements and additions, CPLEX is a powerful and popular solver used to solve LP problems and extensions thereof, such as network flow problems, quadratic programming

problems, quadratically constrained programming problems and mixed integer programming problems [72].

CPLEX employs a robust algorithm based on the celebrated branch-and-cut method (see §2.4.1) to solve an MIP model [71, p.222]. Several parameters of the branch-and-cut method used by CPLEX may be specified by the user [71, p.224] so as to facilitate more effective execution of the algorithm. For example, the user may choose whether the emphasis in the optimisation process should be on feasibility or optimality, without sacrificing the accuracy achieved during the process. Termination criteria may also be set, such as a computation time limit, a limit on the size of the branching tree constructed or a tolerance associated with optimality. CPLEX can generate thirteen types of cutting planes during algorithmic execution, depending on the nature of the model to be solved [71, p.231].

In order to calculate the daily waste values during a decision period, the inventory levels, order quantities, sales data and historical waste values of the previous decision period are required. Further input to the model include the number of products to be displayed and their respective characteristics, the forecasted demand for the current decision period and the amount of shelf space available for the display of fresh produce. The required data are assumed to have been prepared and stored in Excel workbooks which may be imported into CPLEX.

In order to incorporate the input data as part of the mathematical model in CPLEX, the days of both the previous decision period and the current decision period are numbered. Furthermore, five classes of variables are defined as decisions variables. The main decision variables are the integer-constrained order quantities q_{ij} in (3.7)–(3.13), which range over all the products $i \in \mathcal{P}$ and over the days $j \in \mathcal{J}$ in the current decision period. The starting inventory values r_{ij} , the product waste values w_{ij} , the underachievement values σ_{ij} and the overachievement values μ_{ij} are auxiliary variables which depend on the order quantities. The starting inventory and product waste variables are defined over all the products and over all the days of both decision periods. The relevant model input values are then assigned to the inventory and waste variables associated with the days of the previous decision period. The underachievement and overachievement variables are also defined over all the products, but only over the days of the current decision period \mathcal{J} , because no previous values of these variables are required for calculations in support of decisions during the current decision period.

The objective function is implemented in CPLEX according to (3.1), ranges over all the products in \mathcal{P} and all the days of the current decision period \mathcal{J} , and is subject to the constraints (3.8)–(3.11). These constraints are applied over the entire range of products, and over all the days of the current decision period as well as an additional number of days (in order to account for the lead time) so as to ensure that the quantities ordered during the last few days of the current decision period before the start of the next decision period also adhere to the constraints. The code used to formulate the model in CPLEX is provided in Appendix A. Although there exist nonlinearities in the proposed model as a result of the use of the maximum operator, it is converted to an MIP formulation by CPLEX and may therefore be solved by its linear solver [71, p.327].

4.2 Product data for hypothetical test instances

Hypothetical product data are presented in this section which may be used to construct problem instances for demonstration purposes as input to the mathematical model of Chapter 3. Suppose a retail outlet has up to ten products on display in its fresh produce department. These hypothetical products and their hypothetical characteristics are presented in Table 4.1.

TABLE 4.1: Characteristics of products for the hypothetical problem instances considered in this chapter.

	Product	Selling price (R)	Cost price (R)	Shelf life (days)	Lead time (days)	Pack size	Shelf space (cm^2)	Safety stock	Stacking factor
1	Avocados	6.95	6.00	8	3	60	54	45	2
2	Kiwi	5.00	3.75	4	3	100	25	9	3
3	Cucumbers	8.95	8.20	7	3	30	112.5	30	3
4	Pears	16.50	12.00	6	3	12	260	4	2
5	Bananas	15.95	11.65	3	3	14	300	6	2
6	Pineapples	9.99	7.50	6	3	18	49	2	1
7	Gem squash	33.99	25.70	13	3	8	196	3	2
8	Peppers	49.00	36.00	7	3	8	196	2	2
9	Papayas	16.95	12.70	6	3	8	150	3	2
10	Apples	16.50	12.00	6	3	12	151.25	2	2

Products 1, 2, 3, 6 and 9 are sold individually, while Products 4, 8 and 10 are sold per kilogram. Products 5 and 7 are finally sold in larger packets. Relevant product characteristics include the selling price and cost price per product unit, from which the profit per unit may be determined. Recall that the shelf life of a product is the number of days that the product may be displayed before it expires, while the lead time of a product is the number of days that elapse between ordering a product and the arrival of the product at the retail outlet, which is taken as three days for all the products in Table 4.1 in order to simplify the analysis at this stage. Products are ordered in packs and the pack size of a product is the number of product units in a single pack, while the space characteristic in Table 4.1 specifies the number of squared centimetres that one product unit requires for display purposes. The safety stock values are the numbers of product units that should be on display in addition to the product units required to satisfy the demand. Finally, the stacking factor values are the numbers of units of a product that may be stacked on top of each other in display. For example, a display of cucumbers may be three layers high (*i.e.* a stacking factor of 3), while a display of pineapples consists of one layer, with the pineapple leaves pointing upward (*i.e.* a stacking factor of 1).

TABLE 4.2: Product demands for an extended decision period of 34 days, for the first three products in Table 4.1.

Day	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
Avocados	36	46	22	6	29	49	40	37	35	41	41	52	44	67	51	100	108
Kiwi	19	6	8	10	3	14	17	3	0	2	4	13	6	7	8	17	8
Cucumbers	14	37	9	56	30	59	4	100	50	31	30	31	69	75	46	25	35
Day	18	19	20	21	22	23	24	25	26	27	28	29	30	31	1	2	3
Avocados	100	77	72	94	53	84	69	13	11	7	34	60	55	69	60	55	69
Kiwi	24	14	8	3	12	10	0	4	6	14	8	3	12	10	3	12	10
Cucumbers	44	30	30	57	85	40	82	64	39	51	91	87	31	52	87	31	52

As mentioned before, in order to determine a replenishment order schedule for a period of one month, output values of the previous month are required as additional input. For the hypothetical problem instance, product demand values for two decision periods were randomly generated. For the previous month, output values in the form of a replenishment order schedule and daily inventory, underachievement and overachievement values, and waste values were generated in Microsoft Excel to satisfy the demand for the period approximately, without taking (3.7) and (3.8) into account, but adhering to (3.9), (3.10) and (3.11). Some adjustments were made to the replenishment order schedule thus generated in order to ensure that orders placed during the

previous decision period do not result in a violation of the shelf space constraint in the current decision period, especially in cases where the available shelf space is taken as a very limiting value. The demand for the first three products in Table 4.1 is summarised in Table 4.2 over a decision period of 31 days together with an additional three days to account for lead time. The corresponding demand values for all ten products in Table 4.1 are included in Appendix C.

4.3 Numerical results

A sequence of three increasingly complex problem instances are solved in this section, where the number of products increases from one to three in the sequence, and the available shelf space is varied in order to compare the model solutions obtained as well as the corresponding solution times. In the following discussion, output values are analysed, the calculation of waste is explained, and reasons are provided for why the CPLEX output might suggest that no products should be ordered during the decision period. For each solution, the output provided by CPLEX is exported to specified fields in an Excel spreadsheet. The spreadsheet is prepared with tables to which the replenishment order schedule for the decision period, as well as the inventory values, the waste values and the overachievement and underachievement values may be exported. The purpose of this section is therefore both to validate (by inspection of its solutions) the model of Chapter 3 and to ascertain the ability of CPLEX to solve the model within a reasonable time frame.

4.3.1 Solution for the first product in Table 4.1

The model is first solved by CPLEX for the first product in Table 4.1, namely avocados, in isolation. A solution is quickly obtained and the output is simple enough to analyse manually. The sum of the demand and safety stock for avocados fluctuates between 51 and 153 units, and avocados are ordered in packs of 60 units. The amount of shelf space available is chosen as 7000 cm^2 for the first problem instance. The number of packs to be ordered on each day of the decision period, as obtained through CPLEX, may be seen in Table 4.3. The solution is obtained in approximately six seconds on an Intel(R) Core(TM) i7-4790 CPU with 8.00 GB RAM operating at 3.60 GHz within a 64-bit operating system. The inventory levels, the waste values and the under- and overachievement values provided by CPLEX are displayed in Tables 4.4–4.7, respectively. These values may be analysed for verification purposes. The days are numbered from 32 to 65, because the input data are allocated to the first 31 days, which represent the previous decision period, and the current decision period consists of 31 days together with an additional three days to account for lead time.

For example, on Day 42, two packs of avocados are ordered, resulting in 120 units. Because the lead time associated with avocados is three days, the two packs ordered on Day 42 arrive at the start of Day 45 and may be used to satisfy the demand and cover the safety stock of Day 45 onwards. The starting inventory on Day 45 is 46 units, and the arriving units increase the displayed inventory to 166 units. The sum of the demand (67 units) and the safety stock (45 units) for Day 45 is therefore 112 units, which is satisfied by the displayed inventory. Because there is no waste on Day 45, the remaining units at the end of the day amount to 99 units, which will also be the starting inventory for Day 46. Since the minimisation of waste forms part of the objective function, the replenishment order schedule obtained through CPLEX does not result in any product waste during the entire decision period. Some waste was, however, deliberately generated for illustrative purposes as a result of the model input values, and will be

TABLE 4.3: Ordering schedule obtained from CPLEX when solving the model (3.7)–(3.13) for the hypothetical problem involving the first product in Table 4.1 and 7000 cm² available shelf space.

Day	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	
	2	0	1	1	1	0	0	1	1	0	2	0	3	1	2	
Day	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62
	2	0	2	1	2	0	3	0	0	0	0	2	1	1	0	

TABLE 4.4: Inventory values for the decision period obtained from CPLEX when solving the model (3.7)–(3.13) for the hypothetical problem involving the first product in Table 4.1 and 7000 cm² available shelf space.

Day	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48
	0	0	0	0	114	85	96	116	139	104	63	82	90	46	99	48	128
Day	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64	65
	80	100	143	71	97	104	140	71	238	227	220	186	126	71	122	122	127

TABLE 4.5: Product waste values for the decision period obtained from CPLEX when solving the model (3.7)–(3.13) for the hypothetical problem involving the first product in Table 4.1 and 7000 cm² available shelf space.

Day	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Day	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64	65
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

TABLE 4.6: Underachievement values for the decision period obtained from CPLEX when solving the model (3.7)–(3.13) for the hypothetical problem involving the first product in Table 4.1 and 7000 cm² available shelf space.

Day	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48
	81	91	67	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Day	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64	65
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

TABLE 4.7: Overachievement values for the decision period obtained from CPLEX when solving the model (3.7)–(3.13) for the hypothetical problem involving the first product in Table 4.1 and 7000 cm² available shelf space.

Day	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48
	0	0	0	69	40	51	71	94	59	18	37	45	1	54	3	83	35
Day	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64	65
	55	98	26	52	59	95	26	193	182	175	141	81	26	77	77	82	13

TABLE 4.8: Summary of the solution times and objective function values obtained from CPLEX when solving the model (3.7)–(3.13) for the hypothetical problem involving the first product in Table 4.1 and various amounts of shelf space available.

Shelf space	1 000	2 000	3 000	4 000	5 000	6 000	7 000	8 000	9 000	10 000
Solution time	5s	6s	6s	6s	6s	51s	6s	6s	6s	6s
Objective	3 150.20	1 612.15	1 112.45	349.60	263.15	227.05	227.05	227.05	227.05	227.05

discussed later as part of the results obtained by solving a problem instance involving the first two products in Table 4.1.

In the Table 4.6, the underachievement associated with avocados on Days 32–34 is a result of the model input and was induced on purpose, as described earlier. The sum of the safety stock and the demand on Day 32 is 81 units, and because the starting inventory on the day is already zero, and no units arrive on that day, the underachievement is calculated as 81 units. No underachievement exists during Days 35–65, as the shelf space is sufficient. There does, however, exist overachievement on each of the days after the first three days. In fact, this will always be the case, since the underachievement and the overachievement are mutually exclusive, and the only other option is that both have a value of zero, which may happen in the unlikely situation where the displayed product units are exactly equal to the sum of the demand and the safety stock.

The value of the objective function is R 227.05, which may be calculated as follows: The sum of the underachievement for the decision period is 239 units. The profit of selling one unit of avocado is R 6.95 – R 6.00 = R 0.95. There is no waste during the decision period. Therefore, the value of the objective function is R 0.95 × 239 = R 227.05.

Several problem instances were, in fact, solved for the case of the first product in Table 4.1 and a decision period of 31 days, but by varying the amount of available shelf space. The results are summarised in Table 4.8. If the amount of available shelf space is reduced to 1 000 cm^2 , the value of the objective function increases, because then there is insufficient shelf space to satisfy demand and simultaneously maintain the desired safety stock level. As the amount of available shelf space increases, however, the value of the objective function decreases. The best possible result is already attained when the available shelf space is 6 000 cm^2 . As mentioned, the value of R 227.05 may be attributed to the model input, which is fixed for this discussion. A summary of the flow of product units during the decision period that results when the replenishment order schedule of Table 4.3 is implemented, may be seen in Figure 4.1.

The solution time in Table 4.8 is the amount of time that it takes CPLEX to find an optimal solution for each problem instance, rounded up to the nearest second. As the amount of available shelf space increases, the solution time remains fairly constant, except for the instance where 6 000 cm^2 shelf space is available. This longer solution time may be attributed to the fact that the amount of available shelf space is close to sufficient and CPLEX therefore requires additional computational time in order to determine which few order quantities should be sacrificed.

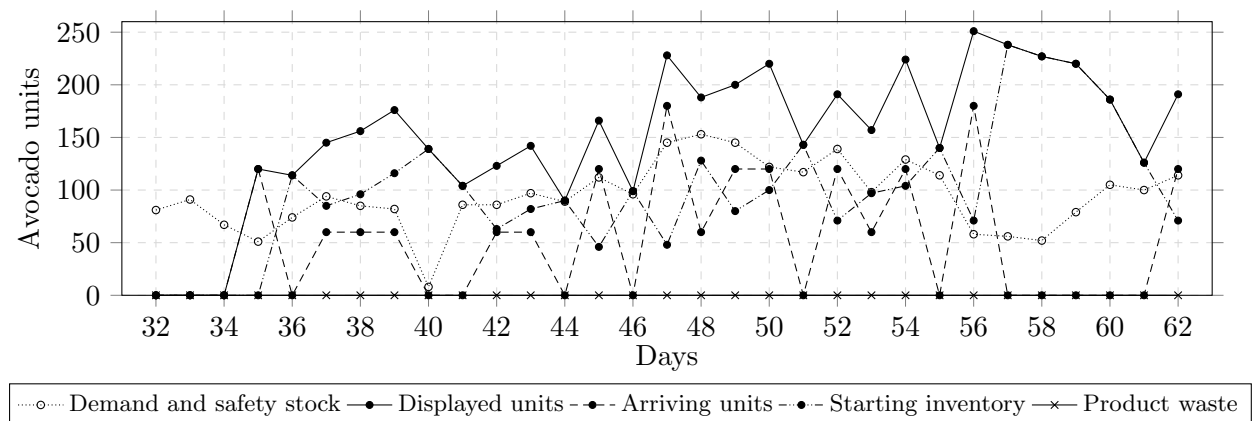


FIGURE 4.1: Summary of the flow of product units during the decision period when 7 000 cm^2 shelf space is available to the first product of Table 4.1.

4.3.2 Solution for the first two products in Table 4.1

Next, the model is solved for the first two products in Table 4.1, namely avocados and kiwi, and the available shelf space is initially set to $8\,000\text{ cm}^2$. The sum of the demand and safety stock for kiwi fluctuates between 9 and 33 units, which is significantly lower than for avocados, and the pack size of kiwi is 100 units, which is the largest of all the products in Table 4.1. Since the shelf life of kiwi is four days, it does not come as a surprise that the solution provided by CPLEX stipulates that no kiwi should be ordered during the decision period, as may be seen in the output summarised in Table 4.9. The number of product units that will be discarded as waste when ordering kiwi far outweighs the loss associated with not satisfying customer demand, due to the combination of the large pack size, the low demand and the short shelf life of kiwi. The inventory levels and the waste values provided by CPLEX are displayed in Tables 4.10 and 4.11, respectively. There is no significance in presenting the underachievement and overachievement values here. The decision period starts with some inventory, which results in overachievement values during the first few days of the decision period, but after it is sold and discarded, the remainder of the days in the decision period have underachievement values that is equal to the sum of the demand on that day and the safety stock value.

Waste exists during the decision period because of the input provided. On Day 27, one pack of kiwi is ordered, which amounts to 100 units. The pack arrives after three days, at the start of Day 30, and is then used to satisfy the demand during Days 30–34. Since the starting inventory of Day 30 is zero, the waste at the end of Day 34 is calculated as $100 - (14 + 8 + 19 + 6) = 53$ units. The selling price of a kiwi is R5.00 and the profit of selling one unit is R1.25. Since the accumulated underachievement over the decision period is 509 units, the contribution of kiwi to the objective function may be calculated as $R1.25 \times 509 + R5.00 \times 53 = R901.25$. The contribution of avocado towards the objective function is exactly the same as in the previous instance considered in §4.3.1, when the model was solved for the first product in Table 4.1, and the total value of the objective function may therefore be calculated as $R901.25 + R227.05 = R1\,128.3$. A summary of several problem instances solved for the first two products in Table 4.1 and with the available shelf space varied in increments of $1\,000\text{ cm}^2$ from $1\,000\text{ cm}^2$ to $8\,000\text{ cm}^2$, is provided in Table 4.12. In most of these problem instances, CPLEX obtains an optimal solution within a few seconds, except for the problem instance when the available shelf space is $5\,000\text{ cm}^2$. For those problem instances where the available shelf space is $6\,000\text{ cm}^2$, $7\,000\text{ cm}^2$ or $8\,000\text{ cm}^2$, the same optimal solution is found. As observed in §4.3.1, the longer solution time associated with $5\,000\text{ cm}^2$ of available shelf space may again be attributed to the fact that the amount of shelf space is nearly sufficient to be associated with an optimal solution, and the additional running time is spent verifying which products should not be ordered on which days.

A summary of the daily shelf space utilisation during the decision period that results when the replenishment order schedule of Table 4.9 is implemented may be seen in Figure 4.2, and the corresponding flow of product units is shown in Figure 4.3.

4.3.3 Solution for the first three products in Table 4.1

To make matters slightly more complicated, the model is now solved for the first three products in Table 4.1, namely avocados, kiwi and cucumbers, and the available shelf is set to increasing multiples of $1\,000\text{ cm}^2$. If the available shelf space is $1\,000\text{ cm}^2$, the optimal solution is obtained in six seconds and the objective function value is R6\,056.95, which is rather high. If, however, the available shelf space is set to $2\,000\text{ cm}^2$, a solution corresponding to an objective function value of R4\,448.90 is obtained in seven seconds. When the available shelf space is further increased

TABLE 4.9: Ordering schedule obtained from CPLEX when solving the model (3.7)–(3.13) for the hypothetical problem involving the first two products of Table 4.1 and 8000 cm² available shelf space.

Day	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47
Avocados	1	1	1	0	2	0	0	2	0	0	2	2	0	2	2	1
Kiwi	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Day	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	
Avocados	2	1	1	2	2	0	0	0	0	1	1	0	1	1	2	
Kiwi	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	

TABLE 4.10: Inventory values for the decision period obtained from CPLEX when solving the model (3.7)–(3.13) for the hypothetical problem involving the first two products of Table 4.1 and 8000 cm² available shelf space.

Day	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48
Avocados	0	0	0	0	54	85	96	56	139	104	63	142	90	46	99	168	68
Kiwi	78	59	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Day	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64	65
Avocados	80	100	83	131	97	104	140	191	178	167	160	126	126	131	62	62	67
Kiwi	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	97	85

TABLE 4.11: Product waste values for the decision period obtained from CPLEX when solving the model (3.7)–(3.13) for the hypothetical problem involving the first two products of Table 4.1 and 8000 cm² available shelf space.

Day	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48
Avocados	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Kiwi	0	53	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Day	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64	65
Avocados	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Kiwi	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

TABLE 4.12: Summary of the solution times and objective function values obtained from CPLEX when solving the model (3.7)–(3.13) for the hypothetical problem involving the first two products of Table 4.1 and various amounts of shelf space available.

Shelf space	1000	2000	3000	4000	5000	6000	7000	8000
Solution time	6s	8s	13s	19s	9min5s	8s	15s	7s
Objective	4051.45	2587.4	2013.7	1312.6	1164.4	1128.3	1128.3	1128.3

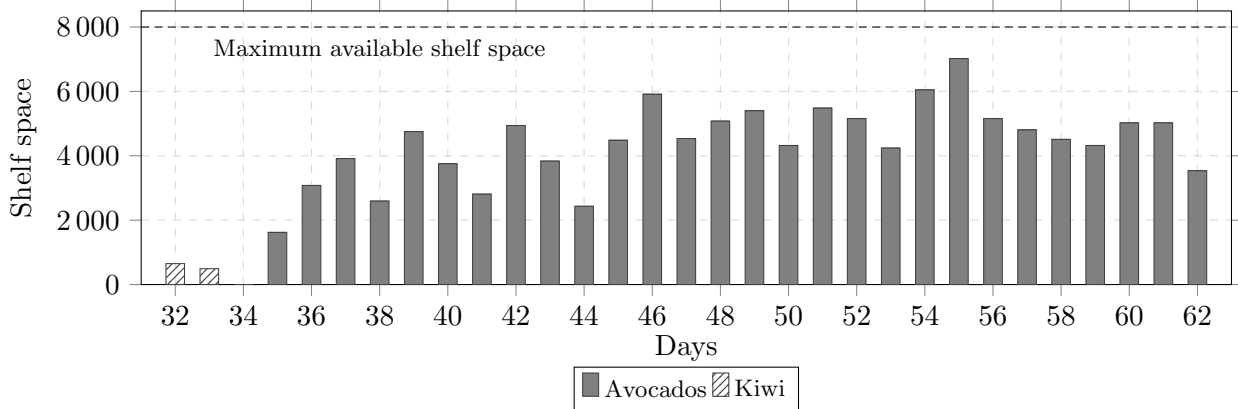


FIGURE 4.2: Daily shelf space utilisation over the decision period for the first two products in Table 4.1 and where 8000 cm^2 shelf space is available.

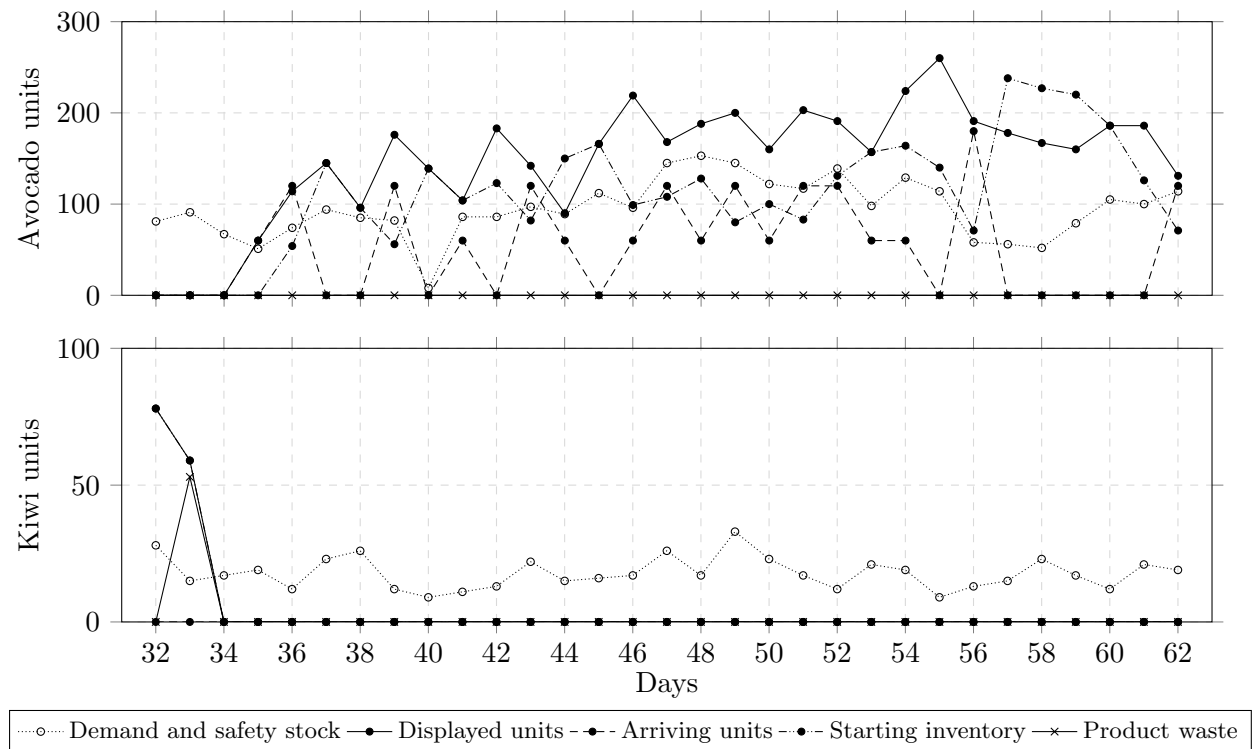


FIGURE 4.3: Summary of the flow of product units over the decision period for the first two products in Table 4.1 and when 8000 cm^2 shelf space is available.

TABLE 4.13: Summary of the solution times and objective function values, and the best bounds and relative gaps, where applicable, obtained from CPLEX when solving the model (3.7)–(3.13) for the hypothetical problem involving the first three products of Table 4.1 and the available shelf space varied.

Shelf space	1 000	2 000	3 000	4 000	5 000	6 000	7 000
Solution time	8s	7s	32s	53min 3s	1 hour	1 hour	1 hour
Objective	6 056.95	4 448.90	3 625.70	2 771.75	2 266.35	1 950.45	1 534.35
Best bound					2 070.92	1 546.2	1 231.28
Relative gap					8.62%	20.73%	19.75%
Shelf space	8 000	9 000	10 000	11 000	12 000	13 000	No constraint
Solution time	1 hour	1 hour	2min 18s	1 hour	9s	1 hour	7s
Objective	1 323.55	1 278.30	1 240.80	1 240.80	1 240.80	1 240.80	1 240.80
Best bound	1 250.18	1 233.30		1 227.05		904.55	
Relative gap	5.54%	3.52%		1.11%		27.10%	

to 3 000 cm^2 , an optimal solution corresponding to an objective function value of R 3 625.70 is obtained in 32 seconds. If the available shelf space is increased by yet another 1 000 cm^2 , the solution time increases significantly, to 53 minutes. For the respective problem instances where the available shelf space is set (in multiples of 1 000 cm^2) from 5 000 cm^2 to 13 000 cm^2 , the solution time is even longer, except for the case where the available shelf space is 10 000 cm^2 or 12 000 cm^2 . A summary of the results obtained for the problem instances solved for the first three products in Table 4.1 is provided in Table 4.13. In the cases where a solution is not obtained after one hour of run time, the solution process is terminated and the results are observed, noting the solution time as one hour.

A useful aspect of the branch-and-cut algorithm is the lower bound on the minimum objective function that it produces. This bound is a value that is deduced as the best possible objective function value for which an integer feasible solution may potentially exist [71, p 223], and is updated as the search for an optimal solution progresses. The *relative gap* associated with a solution is the difference between the current solution and the best bound, normalised by dividing the difference by the best bound together with some tolerance [71, p 284], which is finally given as a percentage. The relative gap may be interpreted as a guarantee on the difference in objective function value between the currently best found solution and an optimal solution [113]. CPLEX provides the best bound and the relative gap for each solution, unless no feasible solutions have been found. It often finds an optimal solution during the early phases of algorithm execution, but then needs significantly more time to adjust the best bound so as to prove that the solution is, in fact, optimal [71, p 263].

The relative gap percentages and best bounds are provided, where applicable, with the associated solutions in Table 4.13. It is interesting to note that in the problem instance where the available shelf space is 12 000 cm^2 , an optimal solution corresponding to an objective function value of R 1 240.80 is once again obtained within a short time, nine seconds. If the available shelf space is 13 000 cm^2 , the optimal solution stays the same, although it is obtained after a far longer time. Upon closer inspection, it may be confirmed that the solution of R 1 240.80 is the best possible solution, since the underachievement associated with kiwi and the historical data are the only contributors to the losses during the decision period reflected in the objective function value.

An interesting experiment may be conducted by eliminating the shelf space constraint and obtaining a solution that is only dictated by the products' shelf lives and demands. The shelf space constraint is the only constraint that links all the products, and therefore it makes sense that CPLEX obtains an optimal solution for three products without any shelf space constraint in a mere seven seconds. The optimal solution to this instance, once again, corresponds to an objective function value of R 1 240.80. The shelf space utilisation does, however, vary remarkably

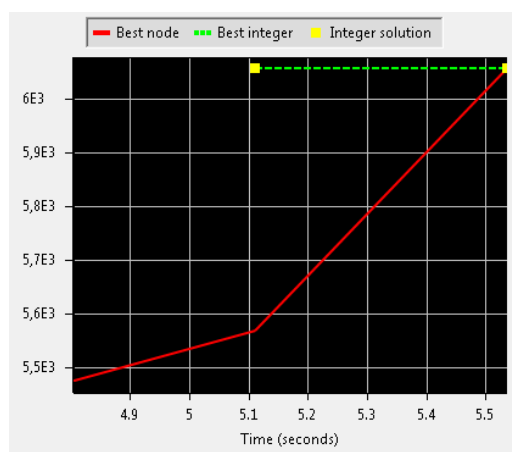
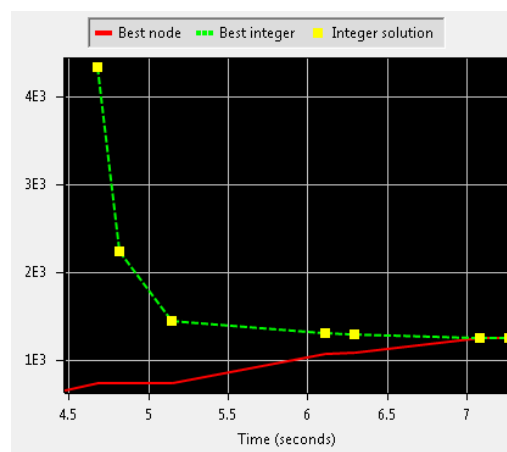
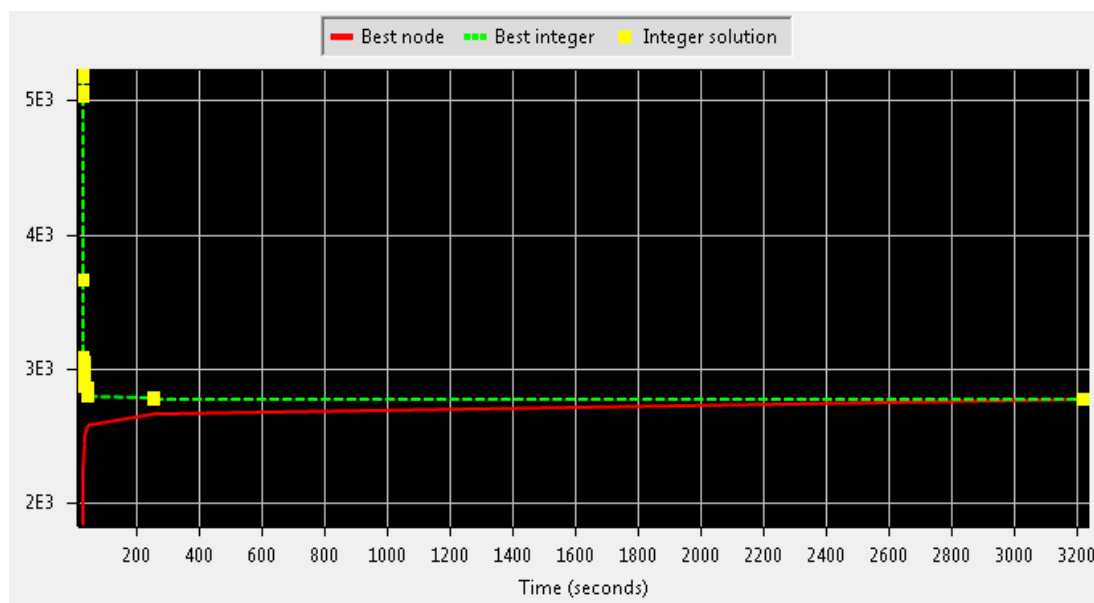
(a) Available shelf space set to 1000 cm²(b) Available shelf space set to 12000 cm²(c) Available shelf space set to 4000 cm²

FIGURE 4.4: Statistics obtained from CPLEX when solving the model (3.7)–(3.13) for the hypothetical problem involving the first three products in Table 4.1. Each graph represents a different problem instance with the respective amounts of available shelf space indicated.

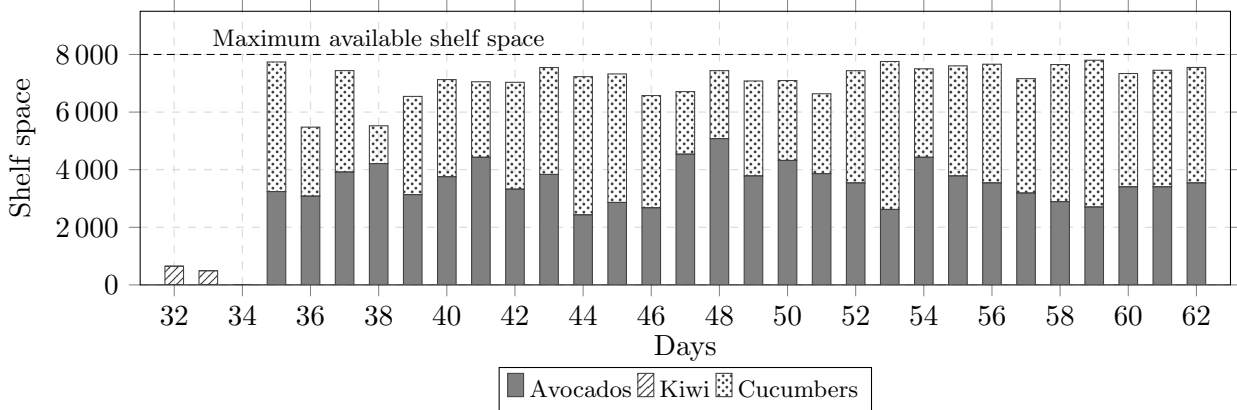


FIGURE 4.5: Daily shelf space utilisation over the decision period for the first three products in Table 4.1 and 8 000 cm² of available shelf space.

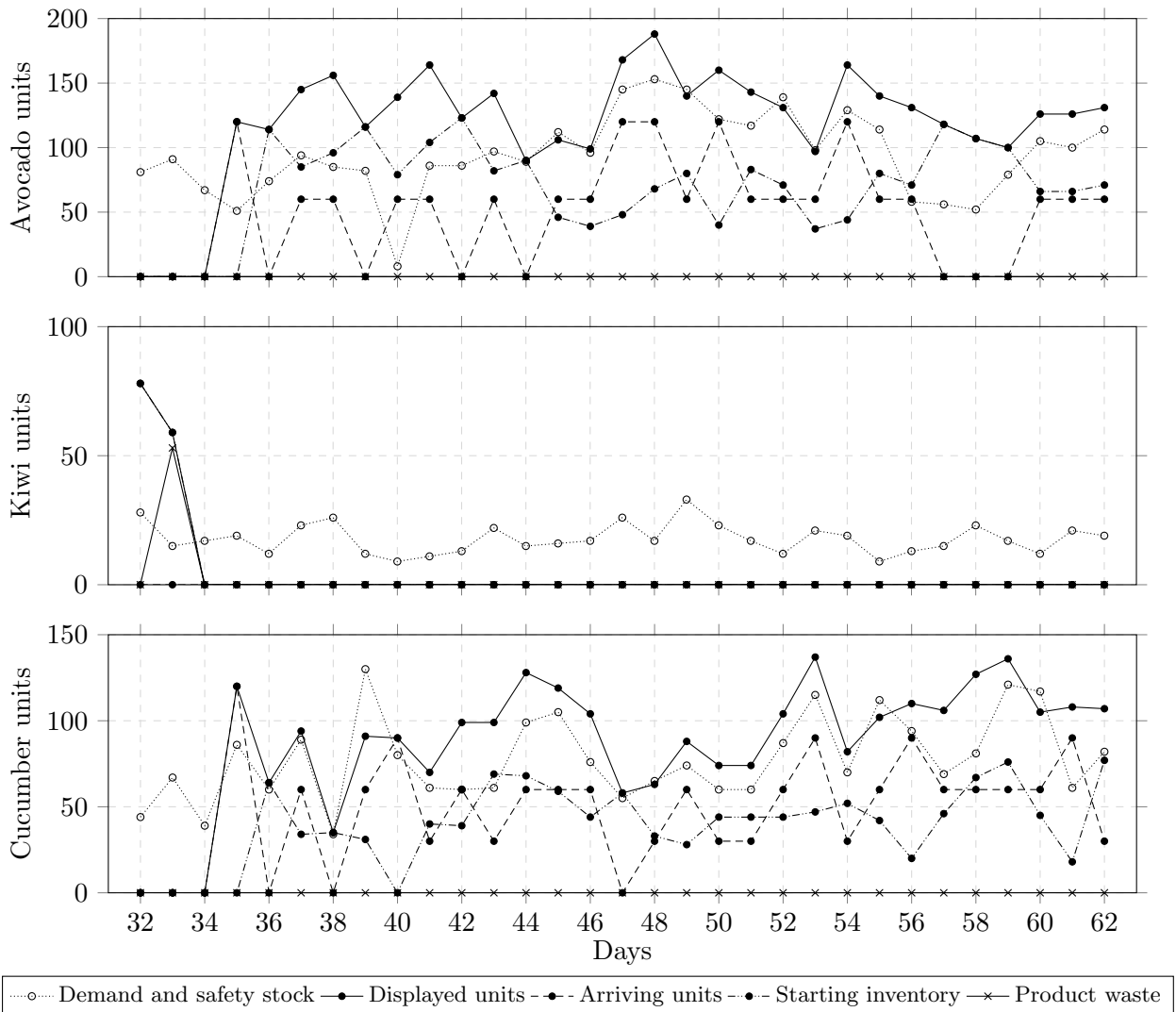


FIGURE 4.6: Summary of the flow of product units over the decision period when 8 000 cm² shelf space is available, for the first three products in Table 4.1.

on the respective days of the decision period as the solution stipulates that large numbers of packs should be ordered infrequently, which is not considered a good solution from a practical perspective. It may be concluded that the reason for the short solution time associated with available shelf space set to $12\,000\text{ cm}^2$, when compared to lower and higher shelf space values, is that the amount of shelf space may be considered sufficient.

Although it may seem that no correlation exists between the solution time required and the solution quality or the available shelf space, a logical explanation may be provided. If the available shelf space is small, it does not take long to confirm that there is insufficient space to satisfy the demand, and an optimal solution, although associated with a large objective function value, is obtained quickly. If more shelf space is available, there are also many more options for replenishment order schedules to explore and compare, which takes time. In such a case, the best lower bound established by CPLEX is implausibly low, especially when the available shelf space is far too much, and it also takes time for CPLEX to adjust the best lower bound. Figures 4.4(a)–4.4(c) contain statistics obtained from CPLEX, depicting the objective function values of solutions and the best lower bound over time. The horizontal axis represents the run time in seconds, and the vertical axis the objective function values. In each of the graphs, the best lower bound is represented by the line that comes from below and increases as it is adjusted with time. The best integer solution found is depicted by the line that comes from above and improves with time as more solutions are explored. Figure 4.4 confirms that CPLEX often discovers the optimal solution early on during a search run, and uses the majority of the run time to update the best lower bound in order to prove that the solution is, in fact, optimal. When the available shelf space is $1\,000\text{ cm}^2$ and $12\,000\text{ cm}^2$, as depicted in Figures 4.4(a) and 4.4(b), CPLEX obtains the optimal solution early on during the search run and the best bound is adjusted quickly. In contrast, when the available shelf space is $4\,000\text{ cm}^2$, CPLEX obtains an optimal solution fairly early during the search run, but spends almost 50 minutes adjusting the best bound, as shown in Figure 4.4(c).

In summary, the daily shelf space utilisation during the decision period that results when implementing the replenishment order schedule obtained from CPLEX, when solving the model for the hypothetical problem involving the first three products of Table 4.1 and $8\,000\text{ cm}^2$ shelf space available, may be seen in Figure 4.5, and the corresponding flows of product units are illustrated in Figure 4.6.

4.3.4 Appraisal of the numerical results

The primary purpose of this section has been to serve as validation of the model derived in Chapter 3 and its solution, as computed by CPLEX. This purpose was achieved through the inspection of the results obtained from CPLEX when solving the model (3.7)–(3.13) for a sequence of small, increasingly more complex hypothetical problem instances. The solution obtained when solving the model for the first product in Table 4.1 was analysed manually in order to verify the values of the variables on individual days over the decision period and to demonstrate how the objective function value is determined. The calculation of product waste was illustrated for the problem instance involving the first two products in Table 4.1, which is in accordance with the waste values obtained by CPLEX. Furthermore, logical explanations were provided for seemingly strange phenomena, such as a replenishment order schedule which stipulates that no product units should be ordered during a decision period. Finally, an interesting experiment was conducted to ascertain the effect of eliminating the shelf space constraint, which is the only model constraint that links the decision variables related to different products.

The secondary purpose of this section was to establish the suitability of the proposed exact solu-

tion methodology implemented in CPLEX. By solving the above-mentioned sequence of problem instances, it was found that the exact branch-and-cut solution methodology employed by CPLEX is very costly in terms of solution time required. Summaries were provided in Tables 4.8, 4.12 and 4.13 of the solutions obtained for several problem instances where the model was solved for the first product, the first two products, and the first three products in Table 4.1, respectively, and in which the amount of available shelf space was varied considerably. From these summaries, it was shown that when only one or only two products are involved, CPLEX solves the model fairly easily by obtaining optimal solutions in short periods of time. When the number of products is increased to three, however, CPLEX struggles to find good solutions within a reasonable amount of time. In conclusion, the quality of the solutions obtained through the employment of the exact solution methodology proposed in the first part of this chapter is therefore considered an insufficient trade-off with the reasonability of the associated model solution time. In the next section, two approximate solution approaches are therefore proposed and applied in an attempt to improve on the practicality of the proposed solution methodology by decreasing the solution time without sacrificing too much in terms of solution quality.

4.4 Solving the mathematical model approximately

The inventory management problem under consideration, fortunately, does not necessarily require an exact solution, since the goal is to satisfy a forecasted demand and not a guaranteed demand. Therefore, an approximate solution is deemed satisfactory, as long as it does not deviate further from an optimal solution than the order of magnitude of possible demand forecast errors. Recall that two categories of approximate solution approaches were discussed in §2.4.2, namely heuristic algorithms and approximation algorithms. The first of two approaches proposed in this section involves imposing a solution time limit on the computations performed by CPLEX, which may be classified as an approximation method because CPLEX provides a measure of sub-optimality, the relative gap, associated with solutions obtained in this way. The second approximate solution approach consists of relaxing the integer constraint so as to obtain a linear programming solution from CPLEX in a very short period of time and rounding the solution to the nearest integers according to certain criteria, which may be classified as a problem-specific heuristic.

4.4.1 Imposing a solution time limit

It was demonstrated in §4.3 that as the number of products considered in the problem, and consequently the number of decision variables, increases, the computational time required by CPLEX increases as it takes longer to verify the optimality of solutions. A computational time limit (measured in seconds) may be imposed on CPLEX to restrict the time it spends searching for an optimal solution. The best solution found up to the point when this time limit is reached is then returned together with the best lower bound and the relative gap percentage available at that point in time. Because the quality of the final solution is provable, the implementation of a solution time limit is classified as an approximation method.

Following on the discussion in §4.3, where an exact solution approach was employed to solve the model (3.7)–(3.13) for hypothetical problem instances (initially the first product in Table 4.1, then the first two products and finally the first three products), the number of products is now increased in order to test the approximation method described above in respect of instances where the first four, then the first five and finally all ten products in Table 4.1 are involved. As in the previous problem instances, the decision period is taken as 31 days and the same data sets

TABLE 4.14: Summary of the objective function values and associated best lower bounds and optimality gaps obtained by CPLEX when solving the model (3.7)–(3.13) for the hypothetical problem instance involving the first four products in Table 4.1 and for three different values of available shelf space, with a computational budget of eight hours imposed.

Shelf space	8 000 cm^2	16 000 cm^2	24 000 cm^2
Objective	1 666.55	1 397.28	1 896.03
Best bound	1 178.77	1 378.11	842.68
Relative gap	29.27%	1.37%	55.56%

are adopted as historical data as were used in §4.3. When the model is solved for the first four products in Table 4.1, with 16 000 cm^2 of available shelf space, without implementing a time limit in CPLEX, the search continues for just over 24 hours without producing an optimal solution. Upon termination of the branch-and-cut search process, CPLEX returns an error message to the effect that the computer ran out of memory.

According to the CPLEX user's manual [71], it is common to run out of memory when solving large MIPs and this occurrence may usually be attributed to the branch-and-cut tree becoming too large. A simple alternative for a user who is satisfied with a solution that is not necessarily optimal, is to impose a time limit on the computation performed by CPLEX. Table 4.14 contains a summary of solutions obtained by CPLEX for the first four products in Table 4.1 when a solution time limit of eight hours is imposed, which is deemed a reasonable time limit for comparison purposes. Because it was previously discovered that the amount of shelf space available affects the solution time, the available shelf space is chosen as three different values, namely a small value that is very limiting, a medium value and a large value that provides sufficient space for all products. The shelf space values are based on the solutions obtained via CPLEX when the shelf space constraint is eliminated. The solutions in Table 4.14 associated with 8 000 cm^2 and 16 000 cm^2 shelf space are obtained when the respective CPLEX search runs reach the solution time limit of eight hours. The solution associated with 24 000 cm^2 shelf space is, however, obtained when the branch-and-cut search terminates after approximately five hours due to a lack of available computer memory.

The effects of the increasing complexity of the problem may clearly be seen when the model is solved for the first five products in Table 4.1 without enforcing a computational budget on CPLEX, with 18 000 cm^2 of available shelf space. The search terminates after approximately six hours because the computer runs out of memory. When compared to the objective function value obtained when the shelf space constraint is eliminated, the solution found when CPLEX terminates after approximately six hours is optimal, but the solver does not have sufficient time to prove optimality of the solution. When the shelf space is 26 000 cm^2 , however, the solution found when the search again terminates after approximately six hours with the same error message, is clearly not optimal, since this solution is worse than when less shelf space is available. When the available shelf space is 10 000 cm^2 , the branch-and-cut search terminates within five hours. A summary of the objective function values and the associated best lower bounds and relative optimality gaps for the three problem instances mentioned is provided in Table 4.15.

Although the time limit approach seems good when solving the above problem instance involving four products, and may be acceptable when solving the instance involving five products, CPLEX struggles to find a solution when the problem instance size is increased to include all ten products in Table 4.1. For all ten products in Table 4.1, the CPLEX search run terminates after eleven and a half hours when the computer runs out of memory when the available shelf space is chosen as the sufficient value of 26 000 cm^2 . The best solution found at termination has an objective

TABLE 4.15: Summary of the objective function values and associated best lower bounds and optimality gaps obtained by terminated CPLEX search runs when solving the model (3.7)–(3.13) for the hypothetical problem instance involving the first five products in Table 4.1 and for three different values of available shelf space, without imposition of a computational budget.

Shelf space	10 000 cm^2	18 000 cm^2	26 000 cm^2
Objective	2 638.32	1 890.76	1 934.73
Best bound	1 189.14	1 302.71	1 306.63
Gap	54.93%	31.10%	32.46%

function value of R 7 166.79, and the relative gap associated with the solution is a large 78.41%. It is clear that imposing a time limit on CPLEX's branch-and-cut search in this instance would have no effect. It may therefore be concluded that the solution approximation approach yields acceptable results when solving the model (3.7)–(3.13) for the hypothetical problem instance involving the first four or five products in Table 4.1, especially since a measure of the degree of sub-optimality is provided, but the approach is not appropriate for larger problem instances. Since real-world problem instances fall in the latter category, other more practical solution approaches should be investigated.

4.4.2 A problem-specific heuristic based on LP relaxation

The complexity of the model (3.7)–(3.13) may, without doubt, be attributed to the fact that the order quantity decision variables are integer-constrained. When the integer constraint is eliminated, the LP relaxation is solved (exactly) in a matter of seconds by CPLEX. The problem-specific heuristic considered in this section therefore involves an elimination of the integer constraint in order to obtain a solution quickly, which may, in turn, be manipulated through rounding in order to deliver an integer-feasible solution.

For comparison purposes, the proposed problem-specific heuristic is applied to obtain a solution to the problem instance involving the first three products in Table 4.1 for three values of available shelf space, namely 1 000 cm^2 , 4 000 cm^2 and 12 000 cm^2 . With the integer constraint relaxed, CPLEX solves all three LP-relaxations in under seven seconds. The solutions to the three problem instances are replenishment order schedules which specify that fractions of packs of each product should be ordered on every day during the decision period, and in the cases where there is sufficient shelf space, the fractions are a combination of values that equal the demand of the day on which the order will arrive and values that will replenish the safety stock. These LP-relaxation solutions are rounded to integers in three different ways in an attempt to obtain good, realistic solutions. Table 4.16 contains an excerpt of the replenishment order schedule obtained from the LP-relaxation solution associated with 12 000 cm^2 shelf space available. Days 32–41 are the first ten days of the current decision period. In order to validate the feasibility of the respective replenishment order schedules, the shelf space utilisation per day is calculated as per (3.2). Shelf space utilisation is significant, because it can be used to compare how well the implementation of different replenishment order schedules will make use of the available resource of shelf space on each respective day during the decision period.

The first method of rounding is referred to as *Rounding heuristic 1*. The type of rounding applied is known as the *floor function* according to which all values are rounded down to the largest previous integer. Table 4.17 contains an excerpt of the replenishment order schedule that results when this method of rounding is applied to the replenishment order schedule for the first ten days of the decision period in Table 4.16. There are many zeros throughout the resulting replenishment order schedule, because the pack sizes are generally large and the demand low

TABLE 4.16: An excerpt of the replenishment order schedule obtained from CPLEX when solving the LP relaxation of the model (3.7)–(3.13) for the hypothetical problem involving the first three products in Table 4.1 and 12 000 cm^2 shelf space available.

Day	32	33	34	35	36	37	38	39	40	41
Avocados	4.50	0	0	0	0	0	0.2	0.68	0.87	0.73
Kiwi	0.44	0	0.03	0.02	0.04	0.13	0.06	0.07	0.08	0.17
Cucumbers	3.87	0	1.97	0.13	3.33	1.67	2.03	0	1.03	2.30

— hence the fractions in Table 4.16 are small. The associated value of the objective function is R 3 457.05, which is relatively large and may be attributed to large underachievement values on the days when no products were ordered to arrive, but demand nevertheless realised. Furthermore, the utilisation of the 12 000 cm^2 available as shelf space is on average less than 50%, which may be regarded as inefficient utilisation of the resource. The last row of Table 4.17 shows the daily shelf space utilisation as a percentage of the total amount of shelf space available. Recall that all three products have lead times of three days, which means that, for example, the product units ordered on Day 32 contribute to the shelf space utilisation on Day 35. The low utilisation percentages on Days 32–34 are a result of the historical data. These findings indicate that Rounding heuristic 1 should be adapted by changing the method of rounding or adding rules to the heuristic.

TABLE 4.17: Excerpt of the solution to the LP-relaxation of (3.7)–(3.13) after applying Rounding heuristic 1 to the replenishment order schedule in Table 4.16.

Day	32	33	34	35	36	37	38	39	40	41
Avocados	4	0	0	0	0	0	0	0	0	0
Kiwi	0	0	0	0	0	0	0	0	0	0
Cucumbers	3	0	1	0	3	1	2	0	1	2
Shelf space utilisation (%)	5	4	0	82	63	57	35	54	27	29

The second method of rounding is referred to as *Rounding heuristic 2* and is the common approach of rounding to the nearest integer. Table 4.18 contains an excerpt of the replenishment order schedule obtained when this method of rounding is applied, as well as the corresponding shelf space utilisation percentages. The resulting value of the objective function is R 1 648.20, which is a considerable improvement on that of the previous heuristic. There are again many zeros in the schedule, for the same reason as mentioned previously. On some of the days during the decision period, the space constraint is violated, which is unacceptable and the solution is therefore deemed infeasible. Furthermore, if, for example, CPLEX were to provide a solution to the LP-relaxation stipulating that 0.5 packs of some product should be ordered on each day of the decision period, each of these values would be changed to 1 after rounding according to Rounding heuristic 2. As a result, double the required amount of packs of the product in question will thus be ordered during the decision period. Based on this brief discussion, there clearly exists a need for further modification of Rounding heuristic 2.

The third method of rounding is referred to as *Rounding heuristic 3* and is a refinement of Rounding heuristic 2. Values in the replenishment order schedule obtained from CPLEX that are below 0.5 are rounded to 0, values that are between 0.5 and 1 are taken as 1, and the rest of the values are rounded according to the floor function. The result of applying Rounding heuristic 3 may be seen in Table 4.19. The value of the objective function associated with this replenishment order schedule is R 2 899.75. Although the solution is an improvement on the solution obtained by Rounding heuristic 1, the space utilisation is still very low, as can be seen

TABLE 4.18: *Excerpt of the solution to the LP-relaxation of (3.7)–(3.13) after applying Rounding heuristic 2 to the replenishment order schedule in Table 4.16.*

Day	32	33	34	35	36	37	38	39	40	41
Avocados	4	0	0	0	0	0	0	1	1	1
Kiwi	0	0	0	0	0	0	0	0	0	0
Cucumbers	4	0	2	0	3	2	2	0	1	2
Shelf space utilisation (%)	5	4	0	92	73	76	46	64	43	38

in the last row of Table 4.19, but at least the solution is feasible in terms of the available shelf space. Rounding heuristic 3 requires further refinement in order to be regarded an acceptable solution method.

TABLE 4.19: *Excerpt of the solution to the LP-relaxation of (3.7)–(3.13) after applying Rounding heuristic 3 to the replenishment order schedule in Table 4.16.*

Day	32	33	34	35	36	37	38	39	40	41
Avocados	4	0	0	0	0	0	0	1	1	1
Kiwi	0	0	0	0	0	0	0	0	0	0
Cucumbers	3	0	1	0	3	1	2	0	1	2
Shelf space utilisation (%)	5	4	0	82	63	57	35	54	27	29

Tables 4.16–4.19 are all applicable to the instance involving three products and where the available shelf space is 12000 cm^2 . The relaxation of the integer constraint and the three rounding heuristics were similarly applied to two other problem instances, where the available shelf space is 4000 cm^2 and 1000 cm^2 , respectively. A summary of the objective function values of all three instances, after applying the three rounding heuristics described above, is provided in Table 4.20 and compared to objective function values when the three instances are solved exactly as MIPs, as was presented in §4.3.3. Although it may at a first glance seem that the objective function values are better in some cases when Rounding heuristic 2 is applied than when the problem is solved as an MIP, the associated replenishment order schedules result in violations of the available shelf space constraint and are therefore infeasible. When the available shelf space is 1000 cm^2 , the replenishment order schedule of Rounding heuristic 1 requires that no products should be ordered during the decision period, because the resource of shelf space is then so limited. After having applied Rounding heuristics 2 and 3, the replenishment order schedules result in undesirable violations of the shelf space constraint. When the available shelf space is 4000 cm^2 , the average space utilisation is low after having applied Rounding heuristic 1. After having applied Rounding heuristic 2, however, the available amount of shelf space is severely exceeded on most days of the decision period. Finally, after having applied Rounding heuristic 3, the average space utilisation is close to 100%, but there are a few days where the available shelf space is still exceeded.

In summary, it is possible to refine Rounding heuristic 3 even further in an attempt to find better solutions that do not violate the shelf space constraint. Too many additional refinements do, however, increase the risk of turning the solution approach into a manual procedure. Therefore, although the rounding heuristics proposed may seem to be an easy way out and are ideal in terms of solution time, the amount of effort required to manipulate infeasible solutions so that they are practically implementable is unreasonable.

TABLE 4.20: A summary of the objective function values obtained after having applied the three methods of rounding to solutions of the model (3.7)–(3.13) for hypothetical problem instances involving the first three products in Table 4.1 with varied available shelf spaces and when solving the problem instances as MIPs. Values typeset in boldface are objective functions corresponding to infeasible solutions.

Space	1 000	4 000	12 000
Rounding heuristic 1	6 121.95	4 934.70	3 457.05
Rounding heuristic 2	4 234.30	2 528.10	1 648.20
Rounding heuristic 3	4 234.30	3 140.00	2 899.75
MIP	6 056.95	2 771.75	1 240.8

4.5 Chapter summary

In this chapter, two solution approaches were described for solving the mathematical model proposed in Chapter 3 — an exact solution to the model and various incarnations of an approximate solution approach. CPLEX was used to solve the model (3.7)–(3.13) exactly as an MIP, using the branch-and-cut algorithm, for hypothetical problem instances involving the first few products in Table 4.1. Based on an appraisal of the numerical results thus obtained, it was concluded that there exists an insufficient trade-off between the quality of the solutions obtained and the required computation time.

Two methods of solving the model approximately were therefore proposed and CPLEX was again used in the implementation of these two methods, namely by imposing a solution time limit (an approximation method), and a combination of solving the LP-relaxation of the model and applying rounding heuristics to obtain integer solutions (problem-specific heuristics). It was found, however, that a more effective solution approach is required to solve problem instances of a higher complexity. The next chapter is therefore devoted to a metaheuristic approach toward finding high-quality solutions to the model (3.7)–(3.13).

CHAPTER 5

Model solution by simulated annealing

Contents

5.1	Algorithm development	65
5.2	Validation of the approach	73
5.3	Solving a larger problem instance	76
5.4	Chapter summary	77

During the appraisal of the exact solution approach and two possible approximate solution approaches in Chapter 4, it was concluded that a more effective solution approach is required according to which the model of Chapter 3 may be solved (approximately). In this chapter, such a metaheuristic solution approach is developed to solve the problem of determining a replenishment order schedule to allocate shelf space to fresh produce in a retail outlet. The method of simulated annealing is selected for this purpose and the application of the algorithm and the finetuning of its parameters to the problem is described in some detail.

The specific implementation of the simulated annealing algorithm is outlined in §5.1. Experiments conducted in the context of small, hypothetical problem instances for comparison with the exact solution approach are summarised in §5.2 in order to validate the simulated annealing approach. Finally, the simulated annealing algorithm is applied to a larger, hypothetical problem instance involving ten products, and the output is analysed in §5.3, so as to assess the scalability of its application in the context of the model of Chapter 3.

5.1 Algorithm development

The method of simulated annealing comprises several vital building blocks, as described in §2.4. The strategy employed for handling constraints is discussed in §5.1.1. A solution construction for initialisation of the search process is proposed in §5.1.2. The move operator used to generate neighbouring solutions and the criteria for accepting neighbouring solutions are explained in §5.1.3 and §5.1.4, respectively. The system temperature depends on the initial temperature, discussed in §5.1.5, as well as reheating and cooling schedules, specified in §5.1.6. Two criteria that should be satisfied to terminate the algorithm are mentioned in §5.1.7. The algorithm pseudocode is provided and interpreted in §5.1.8. Various implementation details are finally provided in §5.1.9.

5.1.1 Constraint handling

The objective in (3.7) is to minimise the underachievement associated with displaying sufficient product units during each day of the decision period, as well as to minimise product waste during the period. Therefore, the cost function employed to determine the suitability or quality of a solution should take into account this underachievement as well as the corresponding waste. During each iteration of the simulated annealing algorithm, a new replenishment order schedule is generated. Assuming that the demand for the period and the necessary historical data are available, the values for the inventory, underachievement, overachievement and waste may be determined from this replenishment order schedule for the entire period.

It is important to note that, from the perspective of the method of simulated annealing, (3.9)–(3.11) are not actually product display constraints, but merely balance equations required to calculate the objective function value. The only real constraint in this respect is (3.8), which specifies the maximum amount of shelf space that may be allocated to fresh produce on any day of the decision period. A penalisation strategy, as described by Talbi [154, p 49], is implemented to handle constraint violations during the search. The cost function value of a solution is calculated by extending (3.1) with the addition of a penalty function. A solution is penalised in proportion to its distance from the feasible region, which may be calculated as the sum of the additional shelf space required during the decision period over and above the amount of shelf space available in order to implement the solution physically on any day of the decision period. After some experimentation, the weight coefficient associated with the constraint violation was chosen as 1. If this weight coefficient value is too large, infeasible solutions are never accepted during the search process, which may cause the search to converge within a seriously sub-optimal feasible region. If, on the other hand, the weight coefficient is too small, the final solution may be infeasible. The penalty function employed may therefore be mathematically expressed as

$$\sum_{i \in \mathcal{P}} \sum_{j \in \mathcal{J}} \max \left\{ 0, \left(\frac{r_{ij} + k_i q_{i,j-\ell_i}}{b_i} \right) e_i - S \right\},$$

where S denotes the amount of shelf space available on any day of the decision period and the other term represents the maximum amount of space required by product $i \in \mathcal{P}$ on day $j \in \mathcal{J}$. More specifically, r_{ij} denotes the starting inventory of product $i \in \mathcal{P}$ on day $j \in \mathcal{J}$, $q_{i,j-\ell_i}$ denotes the ordered units of product $i \in \mathcal{P}$ arriving at the beginning day $j \in \mathcal{J}$ (taking the lead time ℓ_i into account), and k_i , b_i and e_i denote the three product characteristics of pack size, stacking factor and required space per unit, respectively. The calculation of the cost function value may therefore be expressed mathematically as

$$\sum_{i \in \mathcal{P}} \sum_{j \in \mathcal{J}} \left[(p_i - a_i) \sigma_{ij} + p_i w_{ij} + \max \left\{ 0, \left(\frac{r_{ij} + k_i q_{i,j-\ell_i}}{b_i} \right) e_i - S \right\} \right],$$

where σ_{ij} denotes the underachievement of the goal to have a sufficient number of units of product $i \in \mathcal{P}$ on the shelf in order to satisfy demand on day $j \in \mathcal{J}$ and provide for safety stock, w_{ij} denotes the number of units of product $i \in \mathcal{P}$ discarded at the end of day $j \in \mathcal{J}$ as waste, and p_i and a_i denote the unit selling price and the unit acquisition price of product $i \in \mathcal{P}$, respectively.

5.1.2 Construction of an initial solution

An initial solution in the form of a replenishment order schedule is required to launch the simulated annealing algorithm. Because the initial solution has a significant influence on the

effectiveness of the algorithm, it is generated with care. When determining a replenishment order schedule for a decision period, the effect of lead time should be taken into account. Product units ordered on a specific day during the decision period only arrive a few days later, as a result of the lead time of the ordered products. Therefore, forecasted demands for an additional number of days after the decision period also form part of the input to the algorithm. For example, assuming that the maximum possible lead time of a product is three days, the demands for a total of 34 days (from here on referred to as the *extended decision period*) are required for a decision period of length 31 days.

An initial solution may be determined from the forecasted demand for the extended decision period. As a starting point, the total demand for each product during the extended period is obtained simply by adding together the demands during the period for each product, respectively. In addition to this, safety stock of each product is also required during the extended decision period. The safety stock value associated with each product is calculated from the average demand and the lead time, as explained in §2.3.3. Because safety stock is ideally never sacrificed, and generally merely forms part of the display, the total amount of safety stock of a product required during a decision period is dictated by a product's shelf life. When considering the safety stock exclusively, the shorter the shelf life of a product, the more regularly the safety stock should therefore be replenished. Hence the total amount of safety stock required per product during the extended decision period may be calculated by multiplying the safety stock value associated with the product by the number of days during the extended decision period and dividing the result by the shelf life of the product. The total number of product units required per product during the extended decision period may then be calculated by adding the number of units required to satisfy demand and the number of units required for safety stock. The next step in the construction of an initial solution is to calculate the number of packs required per product during each day of the extended decision period, by dividing the total number of products required by each product's pack size and rounding the result up where necessary. These values are referred to as the *ideal number of packs* per product per period. A possible disadvantage of the ideal number of packs per product per period is that a vector of these values for all products may represent an infeasible solution, because of the convention of rounding up to achieve these values, in view of the limited amount of shelf space.

In order to be effective, the initial solution should be feasible and in the form of an integer-constrained replenishment order schedule for the entire decision period so as to satisfy the demand for the extended decision period. Keeping in mind the ideal number of packs per period calculated for each product, an additional separate calculation may be performed to construct the feasible input solution. The forecasted demand for each product over the extended decision period may be used to determine a theoretical, fractional number of packs of each product that is necessary to satisfy the demand on each day. For each product, the demand for each day is divided by the pack size of the product, once again taking the lead time into account. These fractional values are rounded down to determine an integer number of packs of each product to be ordered on each day of the decision period, which is assumed to be 31 days in this thesis, to satisfy the demand over the extended decision period. Although feasible, the resulting replenishment order schedule will, however, not be sufficient to satisfy the demand during the period, because of the convention of rounding down. For each product, the total number of products to be ordered during the period, as stipulated by this replenishment order schedule, may be calculated. The resulting values may be compared to the ideal number of packs per product per period calculated previously, and the differences recorded. As part of the initial solution, the calculated differences are temporarily assigned to an additional fictitious day, resulting in a replenishment order schedule for 32 days. As part of the simulated annealing algorithmic search process, the order quantities assigned to the fictitious day will then be distributed among the

other days. Any solution generated during the algorithm will therefore consist of a replenishment order schedule for 32 days, although the order quantities assigned to the last day are fictitious and are hence not considered in the calculation of the cost function value. When constructing the initial solution as described, an upper bound is set on the total number of units that may be ordered per product during the decision period in any candidate solution considered during the simulated annealing search process. A solution may, however, stipulate that fewer products should be ordered, which means that a large order quantity value will be assigned to Day 32 of the replenishment order schedule.

The process of constructing an initial solution is now summarised by means of an example. Suppose that for a decision period of 31 days, an initial solution is to be constructed for a single product. The pack size of this product is 60 units, the shelf life is 8 days, the safety stock requirement is 45 units, and the lead time associated with the product is three days. An excerpt of the demand for the product during the 34 days of the extended decision period is provided for illustration purposes in Table 5.1.

TABLE 5.1: *An excerpt of the demand for one product over the extended decision period to illustrate the construction of an initial solution.*

Day	1	2	3	4	5	6	...	34
Demand	36	46	22	6	29	49	...	69

Since the lead time is three days, the replenishment order schedule calculated by the simulated annealing algorithm for the number of units to be ordered on each of the 31 days of the decision period will be used to satisfy the demand during Days 4–34. The sum of the demand for the product during Days 4–34 is 1682 units. The number of units that should be ordered to maintain the safety stock level during the extended period is calculated as follows. The length of the decision period (31 days) is divided by the shelf life of the product (8 days) to determine the reorder frequency, and the result is multiplied by the required safety stock level (45 units). This calculation yields a total of 174.38 units that should be ordered for safety stock. Adding the result to the period demand, the total number of units that should be ordered during the decision period is 1856.38. The total number of packs required during the period may therefore be calculated as 1856.38 divided by the pack size (60), which yields 30.94. The result may, in turn, be rounded to 31 packs, which is the ideal number of packs required for the product during the decision period.

TABLE 5.2: *An excerpt of the theoretical, fractional order quantities for one product over the decision period to illustrate the construction of an initial solution.*

Day	1	2	3	...	31
Order quantity	0.10	0.48	0.817	...	1.15

An excerpt of the theoretical, fractional number of packs that is necessary to satisfy the demand during the extended decision period, taking lead time into account, is provided in Table 5.2. The fractional order quantity for Day 1 is calculated by dividing the demand for Day 4 (6 units) by the pack size (60 units), and the same calculation is followed for the other days. In order to generate a feasible initial solution, the fractional order quantities are rounded to the nearest integers, as presented in Table 5.3.

The total order quantity for the decision period, as stipulated by the resulting integer-constrained replenishment order schedule, is 30 packs. When compared to the ideal number of packs calculated previously, the difference is recorded as $31 - 30 = 1$. The final step in the construction of an initial solution is to add a fictitious day to the replenishment order schedule and assign the

TABLE 5.3: An excerpt of the replenishment order schedule for one product over the decision period when the fractional order quantities are rounded, to illustrate the construction of an initial solution.

Day	1	2	3	...	31
Order quantity	0	0	1	...	1

recorded difference as order quantity for the fictitious day. An excerpt of the resulting initial solution is provided in Table 5.4. If further products are involved, the procedure discussed above is applied to each product individually.

TABLE 5.4: An excerpt of the replenishment order schedule for one product, with the addition of a fictitious day, that may be implemented as an initial solution.

Day	1	2	3	...	31	32
Order quantity	0	0	1	...	1	1

5.1.3 The move operator

The move operator also has a significant influence on the effectiveness of the simulated annealing search process, since it ultimately defines the path followed in searching through the solution space. A new neighbour is generated at the start of each iteration as specified by the move operator. The total number of packs to be ordered per product during the entire decision period, together with the fictitious day, depends on the initial solution and remains constant. From the previously accepted solution, one product and two days are chosen at random by the move operator. The move is aimed at *increasing* the order quantity of the selected product on one of these days by one and *decreasing* the order quantity of the product selected on the other day by one. If both randomly selected order quantity values are greater than zero, the order quantity on the first randomly selected day is increased and the second order quantity decreased. If exactly one of the two values is zero, that order quantity is increased (by one) and the other order quantity is decreased. If both order quantities are zero, then no changes are made. The probability of both order quantities being zero is generally low, but the rule is enforced to avoid having negative order quantity values in a solution.

5.1.4 Accepting a neighbouring solution

The neighbouring solution of a current solution generated by the simulated annealing move operator is either accepted or rejected, according to the Metropolis rule [158], as explained in §2.4.2. Based on the cost function values, if the generated neighbouring solution is an improvement on the previously accepted solution, it is accepted (with certainty). Otherwise, if the solution is not an improvement on the previously accepted solution, a random sample from a uniform distribution on the range $[0, 1]$ is compared to the probability

$$p = \exp\left(\frac{Z_c - Z_n}{T_e}\right),$$

where Z_c denotes the cost of the previously accepted solution, Z_n denotes the cost of the neighbouring solution generated by the move operator and T_e denotes the current temperature of the system. A non-improving solution is accepted if the random observation is less than p (in the case of a minimisation problem). If a neighbouring solution has the same cost function

value as the previously accepted solution, it is accepted, since $p = 1$ regardless of the system temperature.

The last accepted solution and its associated cost function value are stored for future comparison purposes, and is updated once the next acceptance is made. When a non-improving solution has been accepted, the probability of finding an improving neighbouring solution, and therefore also the acceptance probability, is higher. The temperature of the system is, of course, also a factor with significant influence on whether or not a solution is accepted.

5.1.5 The initial temperature

The initial temperature T_0 is calculated by implementing a method referred to as the *average increase method* [147, 158]. Starting with the initial solution of §5.1.2, a user-specified number of solutions are generated without accepting or rejecting any of the solutions. This process may therefore be seen as a random walk through the solution space. The cost function value of each solution thus encountered is determined and stored. If the cost function value increases from one solution to the next, the increase is recorded. At the end of the random walk, the average is taken over all these increase values, and this average is referred to as the average increase in energy, denoted by $\overline{\Delta E}^{(+)}$. The initial temperature depends on the average increase in energy and a user-specified parameter, the so-called initial acceptance ratio, denoted here by χ_0 , which is the number of worsening solutions that should be accepted as a percentage of the number of worsening solutions generated. Dréo *et al.* [40, p 45] suggest that χ_0 be assigned a small value, such as 0.2, when the initial solution is considered a relatively good solution, which is the case in this discussion. If the initial acceptance ratio is expressed as

$$\chi_0 = \exp\left(-\frac{\overline{\Delta E}^{(+)}}{T_0}\right),$$

then it follows that the initial temperature may be calculated as

$$T_0 = -\frac{\overline{\Delta E}^{(+)}}{\ln(\chi_0)}.$$

5.1.6 The cooling and reheating schedules

Recall that, in §2.4.2, an *epoch* was defined as a collection of consecutive simulated annealing iterations performed at a constant temperature. An epoch ends by either cooling or reheating the system according to certain criteria, and the length of an epoch is therefore determined dynamically. The celebrated geometric cooling schedule is adopted in this thesis. According to this schedule, the temperature is lowered from epoch e to epoch $e + 1$ by implementing the rule

$$T_{e+1} = \alpha T_e, \quad e = 0, 1, 2, \dots,$$

where α is referred to as the user-specified *cooling parameter*. The value of α typically lies between 0.8 and 0.99 [41]. The temperature is lowered after a pre-specified minimum number of solutions have been accepted; Dréo *et al.* [40, p 45] suggest after $100 \times N$ acceptances. Here N represents the degrees of freedom of the optimisation problem under consideration, which is taken in this thesis as the number of days in the decision period multiplied by the number of products.

Reheating is implemented when no acceptances have occurred over a pre-specified number of iterations. This number is typically taken as $12 \times N$ [40, p 45] iterations. Reheating is also implemented when the value of the cost function remains the same for a specified number of iterations. It frequently occurs that the cost function values of neighbouring solutions are the same in the model of §3.7, due to the nature of the cost function. Because only product waste, underachievement and space violation are measured in the cost function, it makes sense that a small change in the replenishment order schedule will not necessarily result in any change of these three measures. Reheating is implemented a total of at most three times over the course of execution of the algorithm, always as a percentage of the initial temperature. Three reheating percentages, a_1 , a_2 and a_3 , are assigned descending values so as to implement reheating according to a simple schedule. During the first reheat, the temperature is increased to $a_1 \times T_0$, while during the second reheat the temperature is increased to the lower value of $a_2 \times T_0$, and during the last reheat to the even lower value of $a_3 \times T_0$.

5.1.7 Search termination criteria

The simulated annealing algorithm terminates if any one of two stopping criteria is reached (whichever occurs first), namely when a user-specified maximum number of iterations have been performed or when reheating has occurred three times. Therefore, four variables determine the number of search iterations, the number of epochs, the number of reheats and the number of times the same cost function value is found.

5.1.8 Algorithm pseudocode

A pseudocode description of the simulated annealing algorithm implemented in this thesis for computing a replenishment order schedule for the fresh produce department of a retailer is provided in Algorithm 5.1 and is summarised in this section.

The algorithmic parameters that form part of the input are the maximum allowable number of search iterations, the initial temperature, the cooling parameter α , the reheating schedule and the limits against which to evaluate counters that are incorporated to keep track of the number of acceptances, the number of move operator attempts and the number of equal cost function values that may be encountered. The problem specifications refer to the number of products considered, the number of days in the decision period and the amount of shelf space available. The relevant product characteristics, the demand and historical data have been discussed above.

The output is in the form of the best replenishment order schedule found during the course of the search, based on the cost function value, which is also returned. The iteration and epoch counters both start at 1, because the initial solution is already the first iteration which forms part of the first epoch at the initial temperature. Counters for the number of times to reheat, the number of move attempts without any improvement and the number of consecutive iterations at the same cost function value start at zero. The current temperature of the system is denoted by T . In Algorithm 5.1, PreviousQ is the matrix used to store the previously accepted replenishment order schedule from which to generate a neighbouring solution. The `CalculateCost` function consists of several steps required to calculate the cost function value of the current solution, namely determination of the inventory, waste, underachievement and overachievement values for each day of the decision period as well as the amount by which the available shelf space is potentially exceeded. BestSol and BestSolIt refer to the lowest cost function value found thus far during the search and the iteration during which it was found.

Algorithm 5.1: A simulated annealing algorithm for computing replenishment order schedules

Input : Algorithmic parameters, problem specifications, characteristics of products, decision period demand and historical data

Output: A product replenishment order schedule

```

1 ItCounter, EpochCounter  $\leftarrow$  1
2 ReheatCounter, EqualCounter, AttemptCounter  $\leftarrow$  0
3  $T \leftarrow$  InitialTemp
4 PreviousQ  $\leftarrow$  InitialQ
5 CalculateCost(InitialQ)
6 BestSol  $\leftarrow$  Cost
7 BestSolIt  $\leftarrow$  1
8 while ReheatCounter  $\leq$  3 and ItCounter  $<$  MaxIt do
9   ItCounter  $\leftarrow$  ItCounter + 1
10  if Acceptances  $\geq$  AcceptancesLimit  $\times$  EpochCounter then
11     $T \leftarrow \alpha T$ 
12    EpochCounter  $\leftarrow$  EpochCounter + 1
13    AttemptCounter  $\leftarrow$  0
14  else
15    AttemptCounter  $\leftarrow$  AttemptCounter + 1
16  if AttemptCounter = AttemptLimit or EqualCounter = EqualLimit then
17    ReheatCounter  $\leftarrow$  ReheatCounter + 1
18    EqualCounter, AttemptCounter  $\leftarrow$  0
19    if ReheatCounter = 1 then  $T \leftarrow a_1 \times$  InitialTemp
20    if ReheatCounter = 2 then  $T \leftarrow a_2 \times$  InitialTemp
21    if ReheatCounter = 3 then  $T \leftarrow a_3 \times$  InitialTemp
22  CurrentQ  $\leftarrow$  PreviousQ
23  ApplyMoveOperator(CurrentQ)
24  CalculateCost(CurrentQ)
25  if Cost = PreviousCost then EqualCounter  $\leftarrow$  EqualCounter + 1
26  else EqualCounter  $\leftarrow$  0
27  if Cost  $\leq$  PreviousCost then
28    Accept improving solution
29    PreviousQ  $\leftarrow$  CurrentQ
30    if Cost  $\leq$  BestSol then Store best solution
31  else
32    if  $\text{rand}((0, 1)) \leq \exp((\text{PreviousCost} - \text{Cost})/T)$  then
33      Accept non-improving solution
34      PreviousQ  $\leftarrow$  CurrentQ
35  else
36    Reject non-improving solution
37    Cost  $\leftarrow$  PreviousCost

```

While the stopping criteria have not been reached, *i.e.* reheating has not yet been performed three times and the maximum number of iterations has not been reached, the simulating algorithm continues to run. The temperature is decreased by multiplying its value by α if the required number of solutions have been accepted during the current epoch. The AttemptCounter tracks the number of iterations in the current epoch. When it takes too long to affect the required number of acceptances during the current epoch and the number of iterations in the current epoch reaches a pre-specified limit, or the cost function values of a specified number of neighbouring solutions are the same, reheating is performed. By this time the temperature should be quite low, which is expected to be the reason for the low number of acceptances, and reheating occurs according to the reheating schedule stipulated by the percentages a_1 , a_2 and a_3 , as described in §5.1.6.

A neighbouring solution is generated from the previously accepted solution by applying the move operator to the stored replenishment order schedule. The cost function value of the neighbouring solution is calculated, as explained, and then compared to the cost function value of the previously accepted solution in order to determine whether the neighbouring solution should be accepted. If the cost function values are the same, the occurrence is tracked by the EqualCounter. If the new cost function value is lower, it is immediately accepted and also compared to the best solution found so far (updating this best solution found, if necessary). If, however, the cost function value of the neighbouring solution is higher than that of the previously accepted solution, it is accepted with a probability, as described in §5.1.4. Accepted solutions are stored, and the last accepted cost function value is stored as the cost of the current iteration, which is used for comparison purposes during the next iteration.

5.1.9 Implementation details

The simulated annealing algorithm is implemented in the freely available language R, and RStudio is used as the IDE. R was designed primarily for statistical data analysis by Ross Ihaka and Robert Gentleman, whose first names partly influenced the name of the language [65]. The development of R started in 1993 and was influenced by two existing computer languages, one of them S, which also contributed to the name of the new language [73]. RStudio is a free and open-source IDE for R, which was publicly released in 2011 [133]. The source code to the simulated annealing algorithm implemented in this thesis is included in Appendix B.

5.2 Validation of the approach

This section serves as a validation of the simulated annealing implementation of §5.1 as a model solution methodology instead of the exact and approximated approaches of the previous chapter implemented in CPLEX. In Chapter 4, several hypothetical instances of the problem were solved for various combinations of a number of products and the available shelf space in the fresh produce department of a retailer. Although only a few products were considered and the instances were therefore relatively simple, it was demonstrated that CPLEX is not able to solve larger and more complex instances with the required effectiveness. Several experiments are performed in this section to compare the approach of using CPLEX to the simulated annealing algorithm of §5.1 implemented in R, especially in respect of solution time and the quality of the solutions.

Nine problem instances are solved in this section using both CPLEX and the simulated annealing algorithm of the previous section. In addition to solving the problem for the number of products (n) equal to four and five, for which the CPLEX solutions were previously presented

TABLE 5.5: Characteristics of products for the hypothetical problem instances.

	Product	Selling price (R)	Cost price (R)	Shelf life (days)	Lead time (days)	Pack size	Shelf space (cm^2)	Safety stock	Stacking factor
1	Avocados	6.95	6.00	8	3	60	54	45	2
2	Kiwi	5.00	3.75	4	3	100	25	9	3
3	Cucumbers	8.95	8.20	7	3	30	112.5	30	3
4	Pears	16.50	12.00	6	3	12	260	4	2
5	Bananas	15.95	11.65	3	3	14	300	6	2
6	Pineapples	9.99	7.50	6	3	18	49	2	1
7	Gem squash	33.99	25.70	13	3	8	196	3	2
8	Peppers	49.00	36.00	7	3	8	196	2	2
9	Papayas	16.95	12.70	6	3	8	150	3	2
10	Apples	16.50	12.00	6	3	12	151.25	2	2

TABLE 5.6: Shelf space values (in cm^2) for the nine problem instances solved as validation of the simulated annealing approach.

	Small S	Medium S	Large S
$n = 4$	8 000	16 000	24 000
$n = 5$	10 000	18 000	26 000
$n = 6$	12 000	20 000	28 000

in Chapter 4, the problem is now also solved for n equal to 6. In each case, the product characteristics correspond to the first four, the first five or the first six entries of Table 5.5. As in the previous experiments, the amount of shelf space available (S) is chosen as three different values — a small value that is very limiting, a medium value and a large value that provides sufficient space. A summary of the chosen S -values for n equal to 4, 5 and 6 is provided in Table 5.6. The historical data and demand for the current decision period of 31 days are provided in Appendix C.

When solving each of these nine problem instances in CPLEX, the time limit is set to eight hours. In most cases, however, the computer runs out of memory before the time limit is reached, but the resulting solutions are nevertheless retained for comparison purposes. In the simulated annealing algorithm, the maximum number of iterations (MaxIt) is set to 500 000, which also takes approximately eight hours to complete. The initial temperature is calculated according to an initial acceptance ratio $\chi_0 = 0.2$. With slight adjustments to the suggestions by Dréo *et al.* [40, p 45], α is taken as 0.85, AttemptLimit as $50 \times$ the degrees of freedom (number of days in the decision period multiplied by the number of products) and AcceptancesLimit as $12 \times$ the degrees of freedom. Based on experimentation during the development of the simulated annealing algorithm, the values of the other parameters are taken as $a_1 = 0.4$, $a_2 = 0.25$, $a_3 = 0.1$ and EqualLimit = 20. Like CPLEX, the simulated annealing algorithm also terminates before performing the maximum number of iterations in some of the instances, as a result of reheating having been performed three times.

The results obtained for the nine problem instances are summarised in Table 5.7. The CPLEX solutions are better in two of the nine instances, and the simulated annealing solutions in five of the nine instances. In two instances both approaches obtain the same solution. In the instance where $n = 4$ and S is small (8 000 cm^2), when solved by CPLEX, the time limit is reached and the relative gap returned is 29%, but the solution is better than when solved by simulated annealing,

TABLE 5.7: Cost function values of the best solutions found for the nine problem instances by CPLEX and the simulated annealing algorithm (SA).

	Low S		Medium S		High S	
	CPLEX	SA	CPLEX	SA	CPLEX	SA
$n = 4$	1 666.55	2 078.13	1 397.28	1 397.28	1 896.03	1 490.30
$n = 5$	2 638.32	2 270.64	1 890.76	1 890.75	1 934.73	1 890.75
$n = 6$	2 222.21	2 487.68	2 805.59	1 940.64	4 616.02	1 954.79

where the completion of the reheating cycle induced the termination of the algorithm. When S is increased to $16\,000\text{ cm}^2$, however, the same solution is found by both solution approaches. Upon further inspection, when compared to the solution returned by CPLEX for four products without the shelf space constraint, it is found that it is, in fact, the best possible solution associated with $n = 4$, given the input values. When S is set to the large value ($24\,000\text{ cm}^2$), the superior solution is found by simulated annealing, after 182 893 iterations, and for the same instance the CPLEX branch-and-cut procedure terminates before the time limit is reached because of too little memory. The solution produced by the simulated annealing algorithm is certainly not optimal in this case, since it is worse than when less shelf space is available.

When $n = 5$, the best solution is found by the simulated annealing algorithm for all three values of S . When S is set to the medium value ($18\,000\text{ cm}^2$), the best solution is also returned by CPLEX, and it may be confirmed as an optimal solution. In all three cases of S -values, the CPLEX branch-and-cut procedure terminates before the time limit is reached. When S is large ($26\,000\text{ cm}^2$), the solution obtained by the simulated annealing algorithm may also be confirmed as an optimal solution, and is returned within 173 657 iterations.

Finally, when $n = 6$, an optimal solution is not found by CPLEX, although the solution found when S is small ($12\,000\text{ cm}^2$) is better than that returned by the simulated annealing algorithm. When S is, however, set to the medium value ($20\,000\text{ cm}^2$), the superior solution obtained by simulated annealing may, in fact, be confirmed as an optimal solution. Simulated annealing also yields the superior solution, although not optimal, when S is large ($28\,000\text{ cm}^2$).

When looking solely at the solution approach implemented in CPLEX, it is worth mentioning that in all but two of the nine instances, CPLEX terminates after approximately six hours of computation time due to insufficient memory and consequently the time limit of eight hours is never reached. Therefore, in all seven instances where the CPLEX branch-and-cut procedure terminates prematurely, the solutions produced are the best possible solutions that CPLEX may find for each respective instance given the hardware configuration available in this study. The associated relative gaps returned for the solutions in each of these seven instances are on average 50%, and increases as the number of products is increased. Finally, when the available shelf space is large in the cases of four, five and six products, CPLEX never finds an optimal solution.

On the other hand, when only considering the simulated annealing solution approach, it is worth mentioning that for all three values of the number of products, when the available shelf space is large, the algorithm terminates before reaching the maximum number of iterations (500 000). The solutions found in these instances are either equal to or within R100 of the optimal value, as identified when solving the instances without the shelf space constraint. When the available shelf space is, however, set to the medium values, optimal solutions are found when the number of products is equal to four, five and six, and in all cases the algorithm terminates because the maximum number of iterations has been performed, before reheating three times.

Based on the nine instances presented in this section, it may be concluded that simulated annealing is the preferred solution approach. Furthermore, the parameters of the simulated annealing

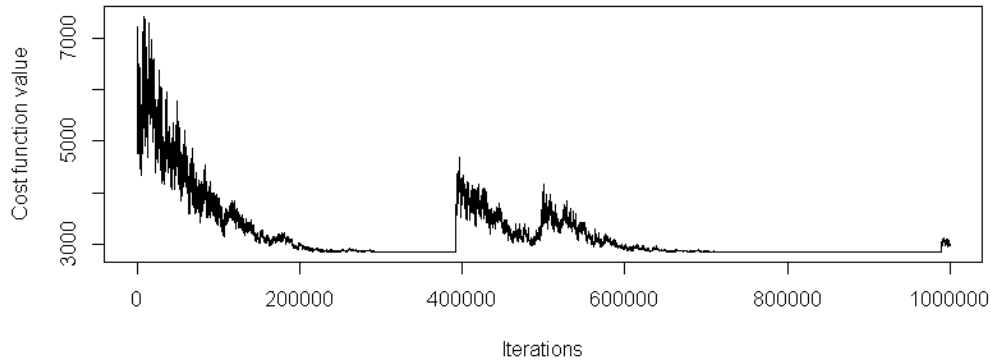


FIGURE 5.1: Cost function values of accepted solutions during the search for the hypothetical problem instance involving all ten products in Table 5.5 with a shelf space constraint of 24 000 cm^2 .

algorithm were kept constant during the instances considered in this chapter, and additional experimentation and modification of the parameters may yield even better results. Finally, it is confirmed that CPLEX struggles more to find optimal solutions as the number of products considered is increased. In summary, the simulated annealing algorithm is now considered validated as the preferred solution approach compared to a solution approach implemented in CPLEX.

5.3 Solving a larger problem instance

The simulated annealing solution approach of §5.1 is applied in this section to all ten products contained in Table 5.5 so as to evaluate the performance of the algorithm in the context of a larger problem instance. The results are analysed to allow for an improved understanding of the algorithm's performance.

Recall that when the instance with ten products ($S = 26\,000\text{ cm}^2$) was solved using CPLEX in §4.4.1, the cost function value of the best solution returned was R 7 166.79, with a relative gap of 78.41%. When the same instance is solved without the shelf space constraint in CPLEX, the objective function value is R 2 837.31, which is therefore a lower bound on the cost function for all products in Table 5.5, and any available shelf space. As discussed before, the solution corresponding to this bound is, however, not feasible, since daily shelf space utilisation is inconsistent and the maximum daily amount of shelf space required is so high that it would violate any reasonable amount of available shelf space in the fresh produce department. The objective function value obtained when the shelf space constraint is eliminated is, however, still meaningful since it may be used as a benchmark against which to evaluate the quality of any solution obtained by the simulated annealing algorithm.

For the larger problem instance, the initialisation of the algorithm is as described in §5.1 and the input data are expanded to include all ten products. The available shelf space is taken as 24 000 cm^2 . The maximum number of iterations are set to 1 000 000 so as to allow for reheating to be implemented more than once. The other parameters remain the same, except for EqualCounter, which is chosen as 22 based on a number of preliminary experimentation runs. The algorithm terminates after the maximum number of iterations have been performed, which occurs after 81 epochs and approximately 22 hours. The best solution found during the algorithmic search has a cost function value of R 2 844.66. This solution is found during iteration

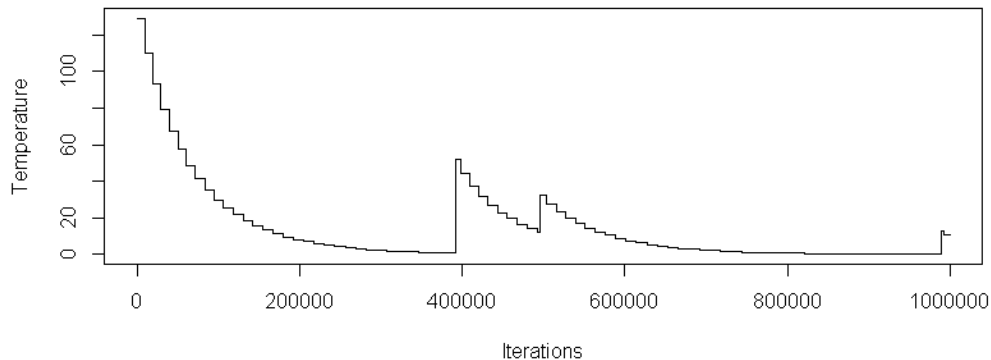


FIGURE 5.2: *The system temperature during the search depicted in Figure 5.1.*

313 397, shortly before reheating is implemented for the first time. This solution is, in fact, very close to an optimal solution, since it is within R8 of the cost function value of the optimal solution obtained when the problem instance was solved without the shelf space constraint (*i.e.* the lower bound mentioned above).

The progression of the search may be seen in Figure 5.1. During the 1 000 000 iterations, reheating is performed three times, although there are insufficient iterations to complete the search cycle after the third reheat. If the maximum number of iterations were specified higher, the search would have continued until just before reheating would have been implemented a fourth time, after which it would immediately have terminated. The change in the system temperature during the search may be seen in Figure 5.2. The initial temperature is calculated by the average increase method, as described in §5.1.5, and taken as 130. Each stage in the graph where the temperature remains constant for a number of iterations represents an epoch.

For this larger problem instance, a near-optimal solution is found, as mentioned. This serves as further validation of the simulated annealing algorithmic implementation and the choice of algorithm parameters. The number of iterations required to obtain the solution may, however, be considered too large. As has been highlighted several times before, an optimal solution (or near-optimal solution) is not required, since the real-life problem under consideration is based on forecasted values.

5.4 Chapter summary

In this chapter, a detailed description was provided of the simulated annealing algorithm employed in this thesis to calculate good replenishment order schedules for the products in the fresh produce department of a retail outlet. The building blocks of the simulated annealing algorithm were discussed, namely the constraint handling strategy, the initial solution and temperature, the move operator, the acceptance and termination criteria, the temperature control schedules, the algorithm pseudocode and, finally, various implementation details. Evidence was provided, in the form of comparative experiments and the solution of a larger problem instance, in support of the validity of the simulated annealing solution approach adopted instead of the exact solution approach of the previous chapter implemented in CPLEX.

CHAPTER 6

Decision support system

Contents

6.1 Decision support system architecture	79
6.2 Decision support system implementation	80
6.3 Chapter summary	86

In this chapter, a *decision support system* (DSS) is designed so as to provide high-level managers of retail outlets with a tool that may be used to calculate order replenishment schedules for fresh produce in line with their retail objectives. In addition, the decision tool may also be seen as a tool to enhance the repeatability and reproducibility of the work performed in this thesis.

The approximate model solution approach of simulated annealing, introduced and validated in Chapter 5, forms the basis of the working of the DSS. The architecture of the proposed DSS is discussed in §6.1 with reference to the three main components of any DSS, namely a database, a *user interface* (UI) and a model base. In §6.2, a concept demonstrator of this DSS is presented by means of a walk-through description of the functioning of its UI.

6.1 Decision support system architecture

A conceptualisation of the envisaged DSS is discussed in this section. The work of the previous chapters is brought together in the design of a DSS process that utilises the method of simulated annealing to compute an order replenishment schedule from user-specified parameters and variables, as well as from input data uploaded by a user.

The main purpose of the DSS is to provide support to a user during the process of calculating order replenishment schedules for fresh produce. It is envisaged that the user of the DSS will typically be a manager of a retail outlet. It is assumed that such a user has access to this thesis and therefore has at least a basic understanding of the working of the simulated annealing algorithm. The DSS may also be used to reproduce the numerical order replenishment schedules discussed in this thesis, since it allows the user to easily change the problem variables and parameters, without having to explicitly make changes to the programming code.

In the previous chapters, parameters were often described as being ‘user-specified’. The DSS facilitates easy specification of these parameters by a user. Furthermore, the length of a decision period, which has up to this point been taken as 31 days, is, in fact, a configurable parameter. Other parameters that may be specified by the user include the available shelf space, the

acceptance percentage and number of iterations used to calculate the initial temperature and associated parameters of the simulated annealing algorithm, such as the maximum number of iterations, the cooling parameter and certain iteration and epoch-related limits. Figure 6.1 provides an overview of the flow of information that forms the architecture according to which the DSS is developed.

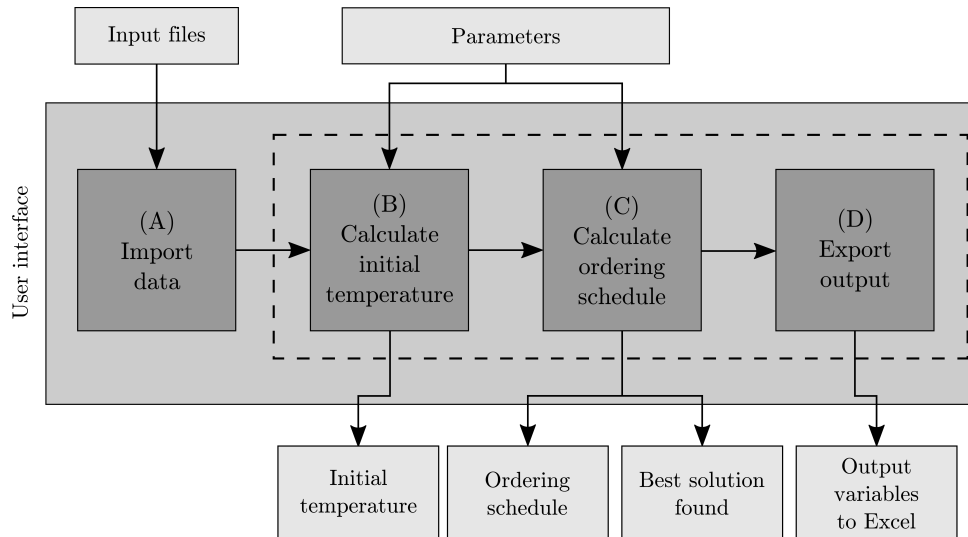


FIGURE 6.1: The flow of information that forms the architecture of the DSS.

The three main components of a DSS were described in general terms in §2.6 — a database, a user interface and a model base. A flat file-based database is selected for inclusion in the envisaged DSS. The input data are uploaded as `.csv` (comma separated values) files, which may be prepared in `Microsoft Excel`. In Figure 6.1, the database is represented by the ‘Input files’ block.

The model base consists of the simulated annealing algorithm proposed in Chapter 5, which is capable of solving the mathematical model derived in Chapter 3 approximately, and is implemented in `R`. The algorithmic parameters are configurable by the user, and have an influence on the time spent computing a model solution and on the quality of this solution. Default values for the parameters are available in case the user does not desire to change any of the parameters. The default values are based on suggestions by Dréo *et al.* [40, p 45]. The two blocks (B) and (C) in Figure 6.1 represent the interaction with the model base.

The UI is designed in `Shiny`, a package in `RStudio` which may be used to build interactive web applications with `R`. The UI is therefore easily compatible with the model base. Furthermore, the `Shiny` software is easy to use and no prior knowledge of web development is required to build a UI using this software. The four blocks (A) to (D) in Figure 6.1 are identified as the main tasks involving the user or for which the user requires assistance.

6.2 Decision support system implementation

The UI has been designed with simplicity in mind, so as not to overwhelm the user. Information displayed on the screen is therefore kept to a minimum. The functioning of the user interface is presented by means of a system walk-through description in this section.

Figure 6.2 contains a screen shot of the main screen displayed when the UI is launched. The

UI consists of two tabs, namely ‘Import files’ and ‘Define parameters & calculate ordering schedule’. These names were chosen to provide the user with a clear understanding of which tasks are performed on each tab. The user may proceed to perform the instructions of the first tab straightaway.

FIGURE 6.2: The main screen displayed when the UI is launched.

The sole purpose of the first tab is to allow the user to upload the .csv files that contain the required input data, namely a list of fresh produce and their characteristics, historical data related to product replenishment decisions during the previous decision period and expected product demand for the current decision period. In Figure 6.3, the user has uploaded three of the eight required files. Each file has to be uploaded separately, and the user is informed to click the ‘Update’ button once all eight files have been uploaded. If this has been performed successfully, the user may proceed to the second tab.

The files that are uploaded here in the system walk-through description correspond to the data provided for the hypothetical instance in §4.2. A replenishment order schedule is therefore determined for a decision period of 31 days and for a fresh produce department where 10 products are displayed.

The screen that appears when the user clicks on the second tab is shown in Figure 6.4. Two main calculations are performed from this screen, and several parameters should be specified in the respective text boxes. Default values are displayed for all but one of the parameters. The first calculation is to determine the initial temperature for the simulated annealing algorithm. Default values are provided for the acceptance percentage and the number of iterations in the random walk, but the amount of shelf space (in cm^2) available for the display of fresh produce should be specified by the user. Prompting occurs in red so as to draw the attention of the user to a text box. Once the available shelf space has been entered, the user may click the ‘Calculate initial temperature’ button. The calculation time depends on the length of the exploratory random walk, possibly specified as a value other than the default 100 iterations by the user.

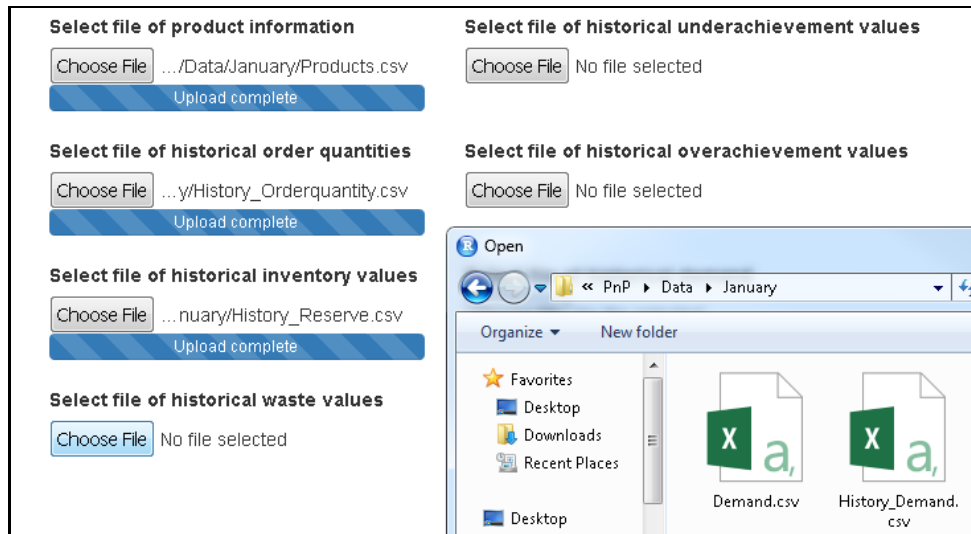


FIGURE 6.3: The main screen of the UI allows the user the upload the required input files.

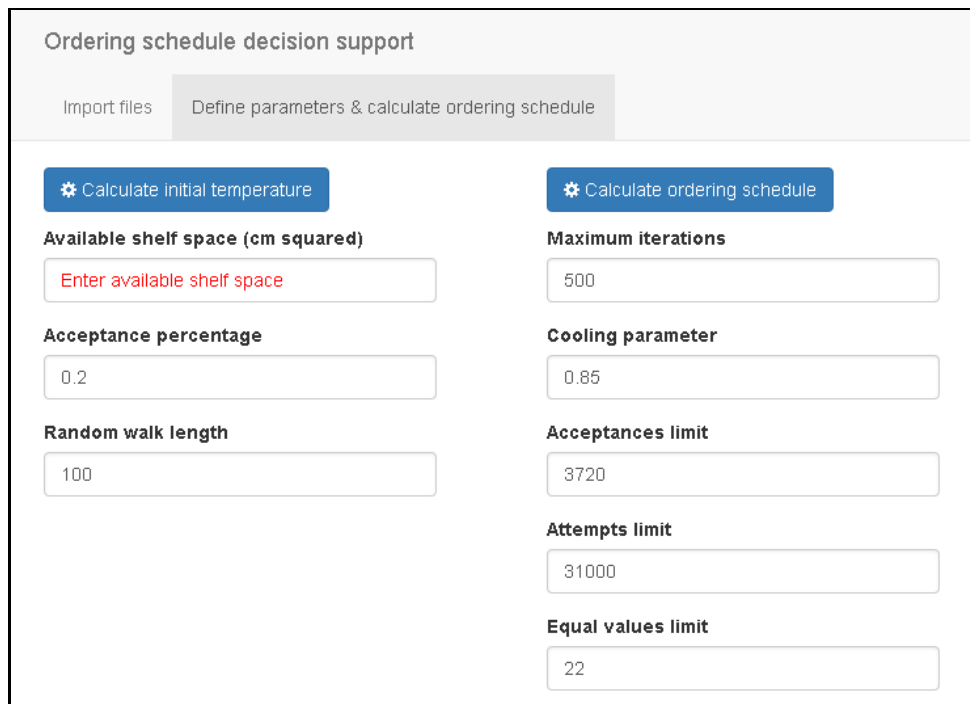
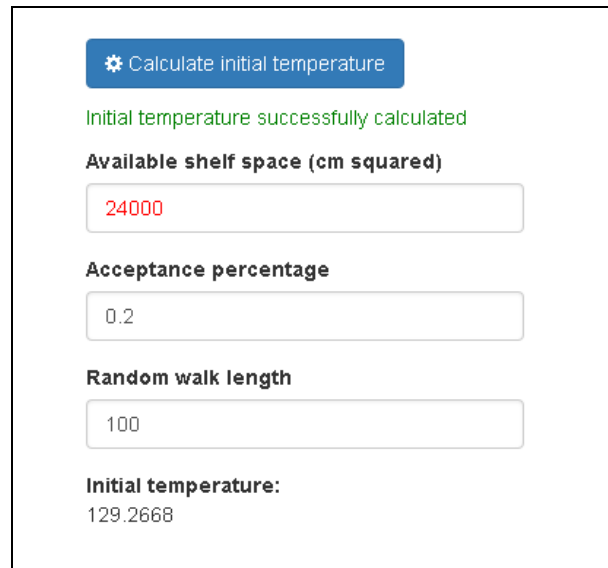


FIGURE 6.4: The second tab is where the user calculates the initial temperature for the simulated annealing algorithm, as well as the replenishment order schedule for the decision period.



⚙ Calculate initial temperature

Initial temperature successfully calculated

Available shelf space (cm squared)

24000

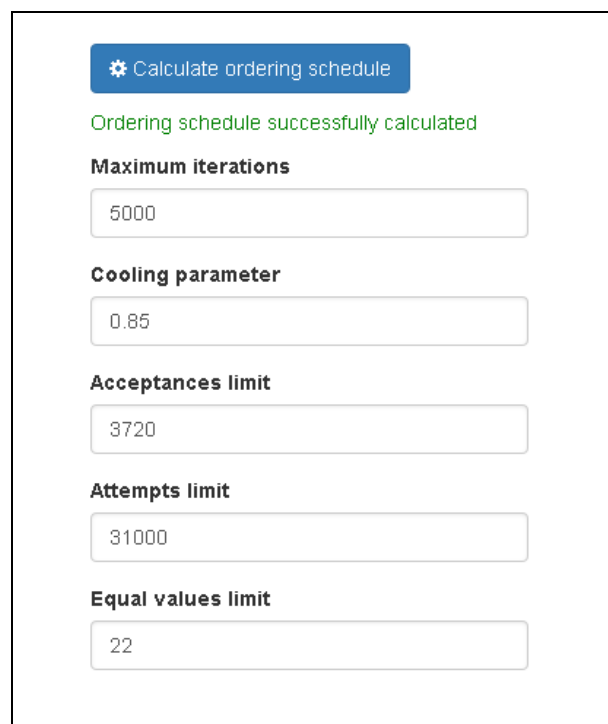
Acceptance percentage

0.2

Random walk length

100

Initial temperature:
129.2668

FIGURE 6.5: *The calculated initial temperature is displayed.*

⚙ Calculate ordering schedule

Ordering schedule successfully calculated

Maximum iterations

5000

Cooling parameter

0.85

Acceptances limit

3720

Attempts limit

31000

Equal values limit

22

FIGURE 6.6: *The user specifies that a maximum of 5 000 iterations may be devoted to searching for a suitable replenishment order schedule.*

Best ordering schedule found

	Day 1	Day 2	Day 3	Day 4	Day 5	Day 6	Day 7	Day 8	Day 9	Day 10
AVOCADO EA	0	4	1	0	1	1	1	0	0	0
KIWI EA	0	0	0	0	0	0	0	0	0	0
CUCUMBERS ENGLISH	3	4	1	3	2	0	0	2	0	4
PEARS PK	0	1	0	0	1	0	1	0	0	0
BANANAS PP	1	0	1	1	0	1	1	0	0	1
PINEAPPLES QUEEN	0	0	0	0	0	0	2	0	1	0
GEM SQUASH LARGE 4EA	0	0	0	1	1	0	1	1	0	1
PEPPERS RED LS PK	0	0	1	0	0	0	0	0	0	0
PAPAYA EA	2	0	0	0	0	0	1	0	0	1
APPLES GR/SMITH 1PK	0	0	0	0	0	0	0	1	0	0

FIGURE 6.7: The best replenishment order schedule found within 5000 iterations is displayed on the screen.

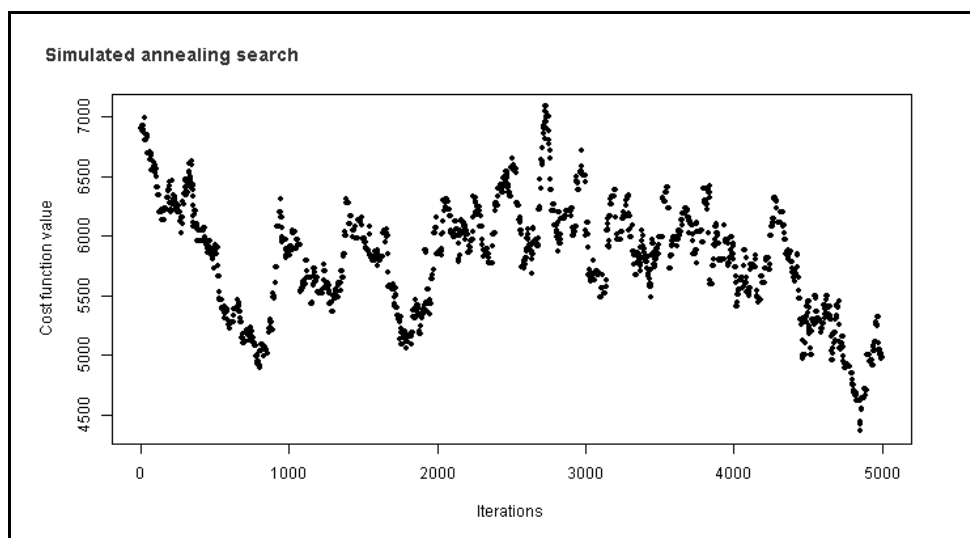


FIGURE 6.8: Cost function values of accepted solutions during the simulated annealing search are presented to the user.

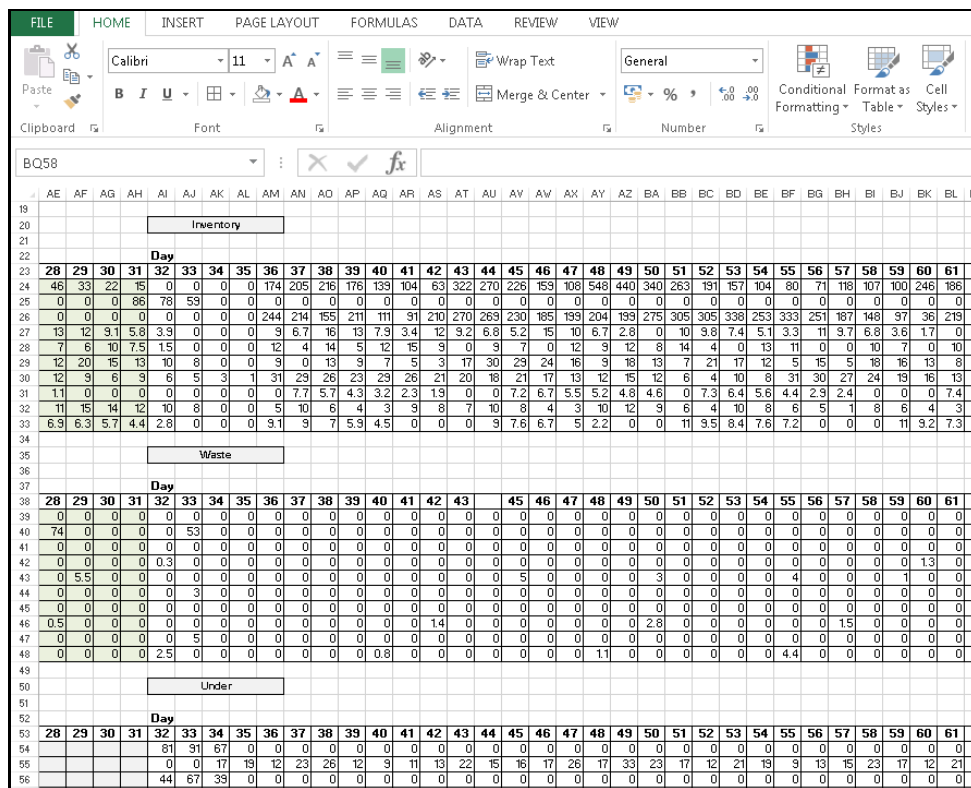
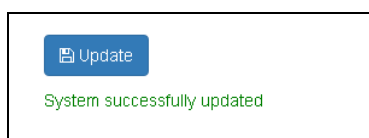
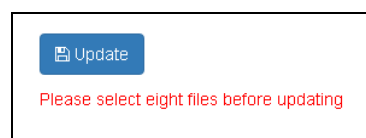


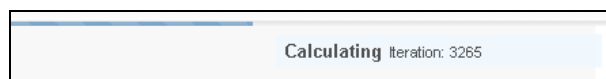
FIGURE 6.9: The output is presented in tables exported to Microsoft Excel.



(a) A confirmation message appears when the input data have been uploaded successfully.



(b) An error message is displayed when the user clicks on 'Update' before uploading all of the required input files.



(c) A progress bar shows the number of iterations that have been performed when the initial temperature is calculated or when the simulated annealing algorithm is running.

FIGURE 6.10: Three additional features incorporated to keep the user informed during the process of calculating a replenishment order schedule for the decision period.

The initial temperature value, together with a confirmation message, appears on the screen as soon as the initial temperature calculation has been completed, as shown in Figure 6.5.

The initial temperature is one of the parameters required to compute a replenishment order schedule. The next step entails the actual computation of the replenishment order schedule. The user may change the default values displayed for the parameters. The default values are calculated at the back-end of the user interface by using information extracted from the input files. As suggested by Dréo *et al.* [40, p 45], the default value of ‘Acceptances limit’ is calculated as $12 \times 10 \times 31 = 33\,720$, and the default value of ‘Attempts limit’ is calculated as $100 \times 10 \times 31 = 31\,000$. The default values for the ‘Cooling parameter’ (0.85) and the ‘Equal values limit’ (22) are based on preliminary experimentation.

In Figure 6.6, it may be seen that the maximum number of iterations has been changed by the user to 5 000, while the other four parameters remain at their default values. Once the user has clicked the ‘Calculate ordering schedule’ button, it takes approximately five minutes to perform the 5 000 search iterations. The user is kept informed on the calculation process by a progress bar that continually displays the number of iterations that have been performed. Once 5 000 iterations have been performed, a confirmation message appears. The simulated annealing search may terminate before the maximum number of iterations is reached, especially when a large value is entered for ‘Maximum iterations’. In either case, the best replenishment order schedule found during the search is displayed, as shown in Figure 6.7. Furthermore, the cost associated with the best replenishment order schedule is displayed below the initial temperature value. In this case, the cost associated with the replenishment order schedule is R 4 361.65. Since 5 000 is relatively few iterations, this cost is rather high.

A graph depicting the progress of the simulated annealing search is displayed below the best replenishment order schedule found. Figure 6.8 shows the graph associated with the search of 5 000 iterations. Each dot in the figure represents an accepted solution. It may be interesting for the user to see how the search progressed, and when the best solution was found. Finally, the replenishment order schedule and the associated inventory values, waste values, and over-achievement and underachievement values may be exported to Microsoft Excel by clicking an ‘Export output to Excel’ button. An example of the output presented in tables in Microsoft Excel may be seen in Figure 6.9.

Additional features were incorporated in the UI in order to keep the user informed during the model solution process. When all eight input files have been uploaded and the user has clicked the ‘Update’ button, the data are saved in R, and the confirmation message of Figure 6.10(a) is displayed. If the user clicks the ‘Update’ button before having uploaded the required files, the error message in Figure 6.10(b) appears. An example of the progress bars that are used to keep the user informed in respect of the number of iterations that have been performed during the calculation of the initial temperature and the replenishment order schedule may be seen in Figure 6.10(c).

6.3 Chapter summary

In this chapter, the work presented in the previous three chapters was drawn together and incorporated in the design of a user-friendly, computerised DSS capable of computing replenishment order schedules for fresh produce in a retail outlet. The proposed components and requirements of the system were first discussed in §6.1. The implementation of the decision support system was presented by means of a system walk-through description in §6.2, and several screen shots were included to illustrate the working of the UI.

CHAPTER 7

A case study

Contents

7.1	Background	88
7.2	Data preparation	88
7.3	Algorithmic parameter evaluation experimental design	89
7.4	Numerical results	96
7.5	Discussion	100
7.6	Chapter summary	104

This chapter is devoted to a case study involving real retail outlet data. The purpose of this case study is to demonstrate that the model of Chapter 3 can be solved approximately by means of the DSS of Chapter 6 in the context of realistic data. The fresh produce department of an outlet of one of the five largest retailers in South Africa, situated in Grassy Park (a suburb of Cape Town), has been identified as an industry partner for the purpose of this case study. The retail outlet exemplifies a relatively new category of stores, referred to as *convenience stores*¹, which are characterised by their small sizes, their limited product ranges and their minimal backroom inventory.

Background information on the case study retail outlet is given in §7.1. A description follows in §7.2 of the data obtained from the retail store and the preparation of the data for the execution of the case study. Varying the parameters of the simulated annealing algorithm of Chapter 5, an experiment is conducted in the context of the case study data in order to determine suitable parameter values for the simulated annealing algorithm. These parameters include the acceptance ratio, the penalty value in the cost function, the cooling parameter, and the parameters *AcceptancesLimit* and *EqualLimit* in Algorithm 2.1. The experimental design is discussed in §7.3, while the numerical results obtained during the experiment are presented in §7.4. The best parameter combination is then employed in §7.5 to solve the model of Chapter 3 for the case study data, and the replenishment order schedule thus obtained is discussed and compared with the replenishment order schedule actually implemented at the participating retail outlet. The chapter finally closes with a brief summary.

¹During the financial year 2015–2016, the retailer in question opened 74 new stores in the form of convenience stores, express outlets, clothing stores and liquor stores [76].

7.1 Background

The convenience store in Grassy Park was opened in January 2015 [153]. In this store, the fruit and vegetables are displayed in crates stacked on the store floor, as well as on refrigerated shelves. The stacked crates are referred to as the ambient section of the fresh produce department, which is also the section considered in the case study. The store serves customers from lower income groups, which is why the store's management team aims to keep operating costs as low as possible. One of the ways in which this is achieved is by displaying fresh produce in the lugs in which they were packaged at the distribution centre, thereby eliminating the costs associated with the additional labour required for unpacking these fresh produce items.

The management team seeks to gain a competitive advantage by improving stock turnover, eliminating (or at least minimising) product waste and not having too much capital tied up in stock. In respect of inventory control, the main focus is on maintaining a suitable safety stock level in order to strive towards avoiding customer disappointment if demand cannot be satisfied. The store's replenishment policy is almost as important as decisions related to the actual display of products in the store. The store operates with minimal backroom inventory. Only 20 percent of the outlet is dedicated to inventory storage, with no backroom space to store additional fresh produce. The entire fresh produce inventory is therefore displayed in the store.

7.2 Data preparation

Three months of data pertaining to fresh produce demand and replenishment decisions were received from the Grassy Park convenience store. These data include a list of all the products sold in the fresh produce department of the Grassy Park retail outlet over a period of three months as well as various characteristics of these products, namely their selling prices, their acquisition costs, their shelf lives, their lead times and their pack sizes. The data furthermore include forecasted daily demands for all the products considered over the three-month period, as well as the actual replenishment order schedule for these products implemented at the store during that time. For discussion purposes, the data may be seen as three separate data sets, namely a set containing the products and their characteristics, a set containing the forecasted demand and the actual replenishment order schedule.

Since it is not available on any of the retailer's systems, physical measurements of the shelf space in the ambient section of the fresh produce department were taken by the author during a visit to the retail outlet in order to calculate the total amount of shelf space available at the outlet. Furthermore, data on the amount of shelf space required to display single product units and the stacking factors associated with these items were also not provided by the outlet. Informed estimates of these data, which have since been validated by a representative of the retailer [153], are therefore employed in the calculations of this case study.

Inevitably, some cleaning and preparation of the data obtained from the outlet were necessary. Only one occurrence of inconsistent spelling of product names was found, and the products are also identifiable by unique product codes. For marketing purposes, some products were sold at a price lower than the acquisition cost, and the associated negative profit values were changed to zero values so that this practice would not have an unnecessary negative influence on the calculation of the cost function values in the model of Chapter 3. In real life, several types of large potato bags are furthermore displayed on the floor of the ambient fresh produce section and not as part of the crate display — the unit space values of these items were therefore taken as zero. Fortunately, the initial solution constructed according to the method described in §5.1.2

will prohibit the model from specifying that unreasonable numbers of these products should be ordered.

Furthermore, the three data sets were analysed to narrow down the list of products for which all the required data are available, since there were some inconsistencies. For example, for one of the products included in the actual replenishment order schedule, no forecasted demand data were available. It might be that this product was ordered as a substitute for another product which could, for some reason, not be ordered. This isolated product was disregarded in the case study. Since only the ambient section of the fresh produce department is considered in the case study, the products were manually categorised into either the ambient or the refrigerated sections. It was found that 72 distinct products are displayed in the ambient section of the Grassy Park convenience store's fresh produce department, while 90 distinct products are displayed in its refrigerated fresh produce section. The former 72 products are considered in this case study. A list of these products and their characteristics, including the characteristics estimated by the author, is provided in Table 7.1. Other input data are included in Appendix D.

The ambient section of the outlet consists of a display of 57 equal-sized crates. The dimensions of a crate are $37\text{ cm} \times 57\text{ cm}$, which results in $2\,109\text{ cm}^2$ of two-dimensional shelf space per crate. The total amount of shelf space available in the ambient section of the fresh produce department may therefore be estimated as $S = 2\,109 \times 57 = 120\,213\text{ cm}^2$. There is, however, also open floor space between and around the crates and it was observed by the author that products are sometimes also displayed on this open floor space. It is therefore possible that the value of $120\,213\text{ cm}^2$ is an underestimate of the actual ambient section shelf space, but this value is nevertheless assumed for the purposes of this case study.

The aim in this case study is to employ the simulated annealing algorithm proposed in Chapter 5 to calculate a replenishment order schedule according to the model of Chapter 3 (by application of the DSS of Chapter 6) for the third month of the period for which data were obtained from the participating retail outlet, in order to compare the schedule recommended by the DSS with the actual replenishment order schedule employed by the outlet over the same period. The data of the first two months of this three-month period were used to obtain a more realistic schedule for the third month. First, the simulated annealing algorithm was used to calculate a replenishment order schedule for the first month of the period taking zero-values as historical order quantities, inventory values and product waste, since no historical data were available for these parameters. Thereafter, this replenishment order schedule and the associated output values were used to calculate a replenishment order schedule for the second month. The resulting replenishment order schedule and associated output values were finally used to calculate a replenishment order schedule for the third month.

7.3 Algorithmic parameter evaluation experimental design

The aim in the experimental design described in this section was to conduct a series of simulated annealing search runs of which the results could be compared in order to find a suitable combination of algorithmic parameter values for solving the model of Chapter 3 in the context of the case study problem under consideration.

Since the problem considered in the case study described in §7.1–7.2 involves many more products than the hypothetical problem instances considered in Chapters 4 and 5, it takes considerably longer to perform an iteration of the simulated annealing algorithm. The investigation therefore leans more towards finding algorithmic parameter values that are likely to decrease the search time, without sacrificing too much in terms of solution quality. The notion of *quenching*,

TABLE 7.1: A list of the 72 products displayed in the ambient section of the Grassy Park convenience store's fresh produce department, together with their characteristics considered in the case study.

Product	Article description	Selling price (R)	Cost price (R)	Shelf life (days)	Lead time (days)	Pack size	Space (cm^2)	Safety stock	Stacking factor
1	CITRUS NAARTJIE 1/2 TRAY	29.95	23.65	11	3	1	600	2	3
2	CITRUS ORANGE 2KG	19.95	16.25	9	3	8	400	2	3
3	KIWI 4EA	23.99	18.90	8	3	8	140	1	6
4	CITRUS ORANGE 1PK	12.95	10.00	11	2	15	384	4	6
5	GEM SQUASH LS 1PK	15.99	13.64	12	3	15	600	7	5
6	BUTTERNUT LS PK	14.99	13.50	12	3	15	250	21	3
7	CITRUS NAARTJIE 1KG	17.95	8.50	11	3	15	400	0	3
8	AVOCADO 4EA	36.95	29.00	6	3	6	400	1	3
9	GEM SQUASH CARRY BAG	38.99	36.37	12	3	1	400	0	2
10	TOMATOES ENGLISH 1KG	17.99	13.15	7	3	14	2500	16	2
11	AVOCADO 2EA	28.95	22.50	6	3	6	200	2	4
12	CUCUMBERS ENGLISH	13.99	10.35	11	3	30	128	41	7
13	PRODUCE PNP SWEET POTATO RED LS PK	17.99	14.50	12	3	18	300	12	4
14	BANANAS PP	21.95	17.20	5	3	14	400	8	4
15	PINEAPPLE QUEEN 2'S	32.99	25.50	10	3	6	450	2	1
16	PAPAYA EA	14.99	11.20	9	3	8	450	3	2
17	ONIONS 2KG	28.99	21.60	12	3	10	1024	7	4
18	CITRUS SOFT CITRUS 1/2 TRAY	44.95	37.00	11	3	1	600	1	3
19	CITRUS ORANGE 4EA	16.95	13.25	11	3	12	324	3	5
20	DRIED CHILLIES 50GR	8.95	6.20	26	4	10	100	1	7
21	CITRUS ORANGE FAMILY POCKET 4.5KG	35.95	32.25	11	3	3	1320	2	2
22	ONIONS RED 1KG	36.99	31.50	12	3	10	400	0	2
23	POTATOES 4KG	45.99	40.23	7	3	1	625	4	1
24	CITRUS LEMON 4'S	28.95	23.00	7	3	8	256	3	5

TABLE 7.1: (continued) A list of the 72 products displayed in the ambient section of the Grassy Park convenience store's fresh produce department, together with their characteristics considered in the case study.

Product	Article description	Selling price (R)	Cost price (R)	Shelf life (days)	Lead time (days)	Pack size	Space (cm^2)	Safety stock	Stacking factor
25	GARLIC LS	149.99	124.50	10	3	10	180	1	6
26	CITRUS NAARTJIE PK	17.95	13.50	11	3	14	360	8	6
27	PRODUCE ONIONS LOOSE 1PK	15.99	11.43	12	3	15	486	47	6
28	CITRUS LEMON PK	35.95	30.00	11	3	15	384	1	5
29	CITRUS SOFT CITRUS PK	0.00	0.00	11	3	14	360	1	6
30	TOMATOES LS 1PK	16.99	12.58	11	3	18	384	27	5
31	P GUAVA PK	0.00	0.00	10	3	6	200	0	5
32	PEARS PK	18.95	14.05	11	2	12	1008	0	5
33	AVOCADO EA	8.49	7.60	10	3	60	100	53	5
34	CITRUS SOFT CITRUS 1KG	29.95	24.50	11	3	15	400	1	2
35	BUTTERNUT CARRY BAG	38.99	33.71	12	3	1	200	1	2
36	KIWI EA	5.99	4.60	10	3	10	35	10	6
37	ONIONS CARRY POCKET	39.99	30.69	12	3	1	400	3	2
38	PRODUCE POTATOES LS CLASS1 MEDIUM PK	13.99	10.06	12	3	15	240	38	6
39	GARLIC CLOVES	17.99	13.60	26	3	25	36	2	6
40	GINGER LS PK	65.99	50.00	10	3	8	600	1	7
41	PINEAPPLES QUEEN	15.99	13.00	7	3	18	225	6	1
42	RED ONIONS LS 1PK	36.99	31.50	12	3	15	567	1	6
43	GARLIC IN NETTING	35.99	27.20	26	3	25	72	2	5
44	GRANADILLA 500GR	19.95	14.30	13	3	6	225	1	4
45	POTATOES BULK BAG 7KG	62.99	51.40	7	3	1	0	2	1
46	BANANAS LS 1PK	12.95	8.80	6	3	17	630	108	6
47	CROUTONS GARLIC & HERBS 55GR	11.99	8.20	17	3	6	120	0	5
48	AVOCADO BACHELOR R & R	15.95	14.00	6	3	8	100	2	6

TABLE 7.1: (continued) A list of the 72 products displayed in the ambient section of the Grassy Park convenience store's fresh produce department, together with their characteristics considered in the case study.

Product	Article description	Selling price (R)	Cost price (R)	Shelf life (days)	Lead time (days)	Pack size	Space (cm^2)	Safety stock	Stacking factor
49	ONIONS DUO PACK	32.99	28.41	12	3	15	160	0	5
50	POTATOES 7KG	49.99	43.58	7	3	1	0	6	1
51	ONIONS LARGE 4EA	19.99	14.80	12	3	12	600	6	4
52	BANANAS BOX 1.2KG	24.95	21.00	5	3	8	540	4	3
53	AVOCADO RIPE & TRIGGERED 6EA	39.95	31.00	6	3	8	600	1	3
54	PEARS KIDS 1KG	14.95	11.25	11	3	16	400	2	2
55	PAPAYA BREAKFAST	39.99	30.50	9	3	6	504	2	3
56	APPLES STARKING 1.5KG	17.99	19.00	11	2	8	700	6	2
57	GEM SQUASH LARGE 4EA	21.99	18.19	12	3	8	225	4	3
58	APPLES FUJI 1.5KG	23.95	17.60	11	3	8	700	1	2
59	APPLES GOLDEN DELICIOUS 1.5KG	17.99	17.80	11	2	8	700	8	2
60	POTATOES WASHED 2KG	28.99	22.35	6	3	8	400	13	1
61	APPLES PINK LADY 4EA	19.99	15.73	6	3	6	169	0	4
62	PAPAYA BABY 2EA	22.99	17.50	9	3	8	300	1	4
63	APPLES KIDS YELLOW 1KG	14.95	13.50	11	3	16	700	6	2
64	APPLES TOP RED 1.5KG	26.95	21.50	11	2	8	700	4	2
65	BUTTERNUTS BAKING	17.99	15.90	12	3	12	250	3	2
66	PEARS PACKHAM 1.5KG	23.95	19.50	11	3	8	700	2	2
67	PEARS 1.5KG	20.95	17.00	11	2	8	700	4	2
68	POTATOES LARGE CHIPS 2KG	28.99	22.35	6	3	8	400	4	1
69	POTATOES SOFT COOKING 2KG	28.99	22.35	6	3	8	400	7	1
70	POTATOES SALAD 1KG	12.99	9.50	6	3	8	400	3	2
71	POTATO BABY 1KG	12.99	9.50	6	3	10	400	6	2
72	SWEET POTATO 1KG	18.99	15.20	9	3	16	400	3	2

explained in §2.4.2, should, however, be avoided.

Throughout the design and implementation of the simulated annealing algorithm and the validation of the algorithm by solving several small problem instances, it was found that some of the algorithmic parameters have a more significant influence on the execution time and the quality of solutions obtained during the search. Recall that the configurable algorithmic parameters are the parameters related to the initial temperature, the maximum number of iterations performed, the cooling parameter, the parameters associated with the reheating schedule, the minimum number of acceptances before the temperature is lowered (*AcceptancesLimit* in Algorithm 5.1), the maximum number of move operator attempts before reheating occurs (*AttemptLimit* in Algorithm 5.1) and the maximum number of equal cost function values that may be encountered (*EqualLimit* in Algorithm 5.1). Changes may also be made to the initial solution and the value with which violations of the shelf space constraint are penalised.

Five parameters were selected for evaluation, based on the perceived significance that changes in these parameter values have on the execution of the simulated annealing algorithm. Two of the parameters may be described as intrinsic parameters in the sense that they affect changes to the algorithm prior to the search. The first intrinsic parameter, the acceptance ratio χ_0 , was selected because of its influence on the initial temperature (an important value). Secondly, the penalty value associated with shelf space violations was selected for evaluation. The other three parameters have a direct influence on the dynamics of the algorithmic search. Since *AcceptancesLimit* affects the simulated annealing epoch lengths, it was also selected for evaluation, while *EqualLimit* was selected because of its influence on the occurrence of reheating. Finally, the cooling parameter α was selected in order to compare different cooling schedules. The evaluation of these five specific parameters covers the different components of the simulated annealing algorithm.

The intrinsic parameters and the parameters influencing the search dynamics were evaluated separately. During the evaluation of the intrinsic parameters, the other three parameters were kept constant, and *vice versa*. By systematically assigning each of the selected parameters a small, a medium, and a large value, in order to form unique parameter value combinations for evaluation, a total of $3^2 + 3^3 = 36$ combinations were formed. Because some of the parameters are kept constant during a portion of the experiments, two of the 36 experiments involve the same set of parameter values, resulting in 35 unique parameter value combinations.

A summary of the small, medium and large values that were assigned to each parameter is provided in Table 7.2. The selected values were mostly based on the recommendations by Dréo *et al.* [40, p 45]. Concerning the parameter χ_0 , it has been suggested that value of 0.2 should be used when the initial solution is assumed to be of good quality, while a value of 0.5 has been suggested for cases where the initial solution is of poor quality. The third value, 0.8, is added for interest's sake. The penalty value was taken as 1 in all the simulated annealing runs described up to now, and so it was thought that it would be interesting to ascertain the effect of penalising infeasible solutions more. Dréo *et al.* suggested a value of $12 \times N$ for *AcceptancesLimit*, where N represents the number of degrees of freedom in the optimisation problem under consideration (taken in this thesis as the number of days in the decision period multiplied by the number of products). Because the case study problem is so large, the adoption of shorter epoch lengths is preferred, which is why the other two values for *AcceptancesLimit* were selected as $6 \times N$ and N . The values for *EqualLimit* are based on preliminary experimentation. For the cooling parameter, Dréo *et al.* suggested a value of 0.9, and two smaller values were also considered to ascertain the effect of cooling the temperature faster.

An extended table containing the values of the five parameters for each of the 35 search runs is

TABLE 7.2: Three different values assigned to each of the selected algorithmic parameters in order to create 35 distinct parameter value combinations for the parameter evaluation experiment.

Parameter	χ_0	Penalty value	<i>AcceptancesLimit</i>	<i>EqualLimit</i>	α
Small value	0.2	1	$N = 2\,160$	15	0.6
Medium value	0.5	10	$6 \times N = 12\,960$	20	0.75
Large value	0.8	100	$12 \times N = 25\,920$	25	0.9

shown in Table 7.3. For the sake of brevity, the search runs were given names, where A and B correspond to the intrinsic parameters, X, Y and Z correspond to the parameters that influence the search dynamics, with numbers indicating the values assigned to the respective parameters.

For practical purposes, it was decided to limit the number of search iterations to at most 300 000, which corresponds to approximately 33 hours of computation time per simulated annealing search run. It is possible that a run terminates before the maximum number of iterations has been executed. Only one search run involving each combination of parameter values was performed, since the search starts from a carefully constructed initial solution, and not from a random initial solution. There is, however, still some randomness involved in the search as a result of the stochastic nature of the simulated annealing move operator and random values incorporated in the decisions related to the acceptance of non-improving solutions. In addition to the best solution found during a search run and the iteration during which it was found, relevant statistics associated with each run were also recorded, namely the number of search iterations executed in total, the total execution time expended during a search run, how many times reheating occurred during a search run and the reasons for reheating. After having completed the 35 search runs, a number of additional runs were performed. More specifically, the amount of available shelf space was varied and the fresh produce replenishment problem was solved with the best combination of parameter values in order to ascertain the effect of available shelf space on the cost function value.

In order to prepare for the first nine search runs involving the evaluation of the value combinations for the intrinsic parameters, a unique initial temperature was calculated for each of the search runs (since the initial temperature is influenced by both the acceptance ratio and the penalty value). The values of *AcceptancesLimit* and α were kept constant for these nine search runs at values suggested by Dréo *et al.* [40, p 45], namely $12 \times N = 25\,920$ and 0.9, respectively. For the next 26 search runs involving the evaluation of the three parameters influencing the search dynamics, the constant values assigned to χ_0 and the penalty parameter corresponded to the best parameter values emanating from the first nine runs.

When analysing the results of the 35 search runs described above, it is important to bear in mind that the historical data and the amount of available shelf space have an observable influence on the resulting cost function values. Although care was taken during the preparation of the data and the related calculations, it is acknowledged that the historical data and the amount of available shelf space may be an inaccurate representation of the real-world problem instance. The input data were, however, identical for all the search runs and for the comparison with the replenishment order schedule actually employed by the retailer. The focus should, therefore, be on the improvements in the cost function values achieved by the simulated annealing algorithm over the course of each search run and on the relative differences between the results of the various runs, rather than on the actual values of the cost function in each case.

TABLE 7.3: The values of the five parameters for each of the 35 search runs in the simulated annealing parameter evaluation experiment.

No	Name	Acceptance ratio	Penalty value	<i>AcceptancesLimit</i>	<i>EqualLimit</i>	Cooling parameter
1	A1B1	0.2	1	$12 \times N$	20	0.9
2	A1B2	0.2	10	$12 \times N$	20	0.9
3	A1B3	0.2	100	$12 \times N$	20	0.9
4	A2B1	0.5	1	$12 \times N$	20	0.9
5	A2B2	0.5	10	$12 \times N$	20	0.9
6	A2B3	0.5	100	$12 \times N$	20	0.9
7	A3B1	0.8	1	$12 \times N$	20	0.9
8	A3B2	0.8	10	$12 \times N$	20	0.9
9	A3B3	0.8	100	$12 \times N$	20	0.9
10	X1Y1Z1	0.2	1	N	15	0.6
11	X1Y1Z2	0.2	1	N	15	0.75
12	X1Y1Z3	0.2	1	N	15	0.9
13	X1Y2Z1	0.2	1	N	20	0.6
14	X1Y2Z2	0.2	1	N	20	0.75
15	X1Y2Z3	0.2	1	N	20	0.9
16	X1Y3Z1	0.2	1	N	25	0.6
17	X1Y3Z2	0.2	1	N	25	0.75
18	X1Y3Z3	0.2	1	N	25	0.9
19	X2Y1Z1	0.2	1	$6 \times N$	15	0.6
20	X2Y1Z2	0.2	1	$6 \times N$	15	0.75
21	X2Y1Z3	0.2	1	$6 \times N$	15	0.9
22	X2Y2Z1	0.2	1	$6 \times N$	20	0.6
23	X2Y2Z2	0.2	1	$6 \times N$	20	0.75
24	X2Y2Z3	0.2	1	$6 \times N$	20	0.9
25	X2Y3Z1	0.2	1	$6 \times N$	25	0.6
26	X2Y3Z2	0.2	1	$6 \times N$	25	0.75
27	X2Y3Z3	0.2	1	$6 \times N$	25	0.9
28	X3Y1Z1	0.2	1	$12 \times N$	15	0.6
29	X3Y1Z2	0.2	1	$12 \times N$	15	0.75
30	X3Y1Z3	0.2	1	$12 \times N$	15	0.9
31	X3Y2Z1	0.2	1	$12 \times N$	20	0.6
32	X3Y2Z2	0.2	1	$12 \times N$	20	0.75
33	X3Y3Z1	0.2	1	$12 \times N$	25	0.6
34	X3Y3Z2	0.2	1	$12 \times N$	25	0.75
35	X3Y3Z3	0.2	1	$12 \times N$	25	0.9

7.4 Numerical results

The numerical results obtained during the 35 search runs of Table 7.3 are presented in this section. In order to interpret the numerical results, two benchmark solutions are presented in §7.4.1, namely the initial solution constructed according to the method described in §5.1.2 and the replenishment order schedule actually employed by the retailer during the last month of the three-month period for which data were available. Results pertaining to the evaluation of the intrinsic parameters are presented in §7.4.2, while the results of the 26 search runs performed to evaluate the parameters that influence the dynamics of the simulated annealing search are presented in §7.4.3. A number of additional search runs were performed to evaluate the effect of increasing the available shelf space, and the results of these search runs are finally presented in §7.4.4.

7.4.1 Benchmarks

The first benchmark solution against which the replenishment order schedule recommended by the DSS of Chapter 6 may be compared, is the initial solution. The same initial solution, constructed according to the method described in §5.1.2, was used to initiate each of the 35 search runs described in Table 7.3. This initial solution is in the form of a replenishment order schedule for the 72 products over the period of 30 days together with an additional fictitious day, and may be found in Appendix D. The cost function value associated with this replenishment order schedule, when the input data is exactly the same as for the 35 search runs, is R 2 907 148. As a replenishment order schedule, the initial solution is in actual fact not a high-quality solution. This may be attributed to the fact that it does not account for safety stock. Recall that a fictitious day is added in the replenishment order schedule to increase the total monthly order quantities of the respective products so as to also provide for safety stock, as described in §5.1.2.

The second benchmark solution is the replenishment order schedule that was actually employed by the participating retail outlet over the same period. The data set was not obtained in the same format as the output of the simulated annealing algorithm, but could easily be manipulated to that format. This replenishment order schedule may also be found in Appendix D. The cost function value associated with this schedule, with the same input data as before, is R 1 789 284.

7.4.2 Results obtained during the first nine search runs

The first nine search runs were performed to evaluate the influence on the solution quality of two selected intrinsic parameters. The results obtained are not of high quality, but were nevertheless deemed a satisfactory foundation for the remainder of the search runs to be performed. The cost function value of the first benchmark solution (the initial solution) was used to normalise the numerical results. Each of the cost function values obtained as results were divided by the cost function value associated with the initial solution. It has been mentioned that the initial solution does not yield a high-quality cost function value, but since each of the search runs started with this same solution, it is a useful indicator of the improvement achieved by the respective parameter value combinations and thus a suitable basis for comparison.

The normalised cost function values of the best solution found during each of the nine search runs are summarised in Figure 7.1. For Runs 1, 4 and 7, the penalty value was 1, while for Runs 2, 5 and 8, it was 10. For Runs 3, 6 and 9, it was 100. The idea behind increasing the penalty value is to decrease the number of infeasible solutions that are accepted and to thereby limit the search to a predominantly feasible search space. The drawback of avoiding infeasible

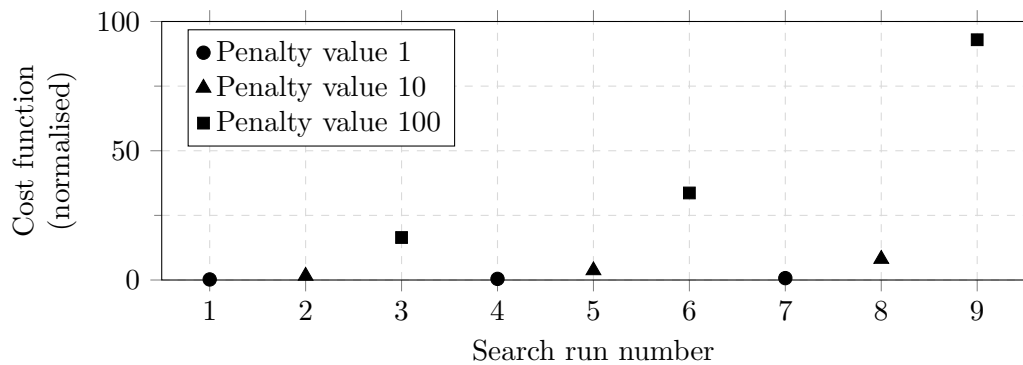


FIGURE 7.1: Normalised results of the nine search runs performed to evaluate the influence of two selected intrinsic parameters.

solutions is that it may close down a direction of search which is the only way of reaching a subset of high-quality solutions. Furthermore, it is clear from the results in Figure 7.1 that higher penalty values have an unfavourable effect on solution quality in the problem instance considered in this case study. This may be attributed to the fact that the amount of available shelf space is particularly limiting, and so the contribution of the daily shelf space violation towards the cost function value is substantial.

The value for the acceptance ratio parameter, χ_0 , was 0.2 for the first three search runs, 0.5 for the middle three search runs and 0.8 for the last three search runs. A lower acceptance ratio yields a lower initial temperature, which implies that non-improving solutions are accepted with a lower probability and that the search progress is biased in favour of improving solutions. This is evident in the results shown in Figure 7.1, where the first three search runs are the set of search runs with the lowest cost function values, taking the influence of the penalty values into account.

The best combination of intrinsic parameter values was found to be those of Run 1, namely when $\chi_0 = 0.2$ and the penalty value is 1. These two parameter values were therefore kept constant for the remaining 26 search runs. The cost function value associated with the best replenishment order schedule found during this search run is R 633 633.84 (a 64.4% improvement in respect of the quality of the replenishment order schedule actually implemented at the participating case study outlet).

7.4.3 Results obtained during the subsequent twenty six search runs

The initial solution was once again used to normalise the results obtained from the 26 search runs involving the simulated annealing parameters which influence the search dynamics. These normalised results are shown in Figure 7.2. A black marker indicates that reheating occurred three times during the associated search run, and the search run terminated as soon as the criteria were met for reheating the system a fourth time. A grey marker, on the other hand, means that the associated search run terminated because the maximum number of iterations had been performed.

It is clear that the grey markers mostly correspond to the higher quality results, while the lowest quality results mostly correspond to black markers. This indicates that reheating was implemented prematurely in the runs associated with the low quality solutions. Upon further inspection, it may be found that there is a common factor among the nine worst black solutions — the value of the *EqualLimit* parameter is 15. The *EqualLimit* parameter has a significant

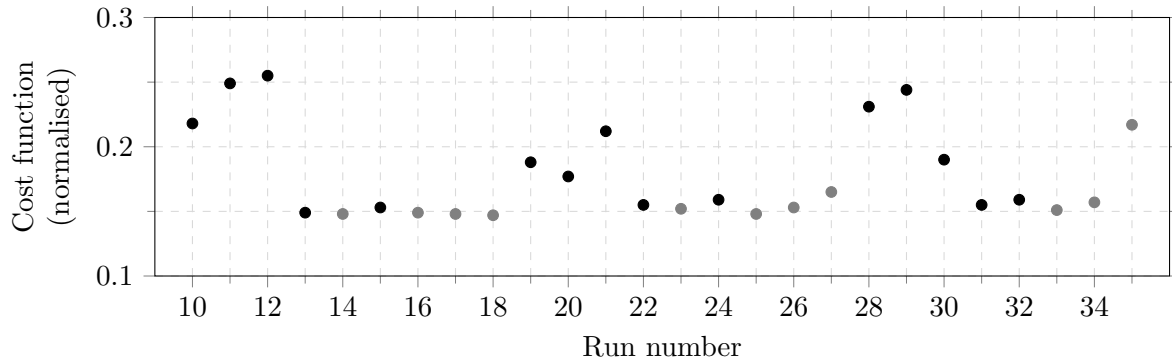


FIGURE 7.2: Normalised results of the 26 search runs performed to evaluate the three selected parameters that influence the simulated annealing search dynamics.

influence on the reheating schedule. Recall that reheating is implemented when neighbouring solutions have equal cost function values for a consecutive number of iterations that corresponds to the *EqualLimit* parameter. A larger value for this parameter might therefore postpone reheating considerably, since it is less probable to achieve a state where the cost function values do not change for a larger number of iterations, and when a change in cost function values does occur, the counting restarts from zero. It may be concluded that assigning a small value to the *EqualLimit* parameter is undesirable.

The *EqualLimit* parameter is responsible for another trend in the results. The termination of the search runs associated with an *EqualLimit* value of 25 may be attributed to the fact that the maximum number of iterations had been reached, and not because reheating had been implemented for the maximum number of times. This confirms the finding above, namely that the *EqualLimit* parameter has a significant influence on the reheating schedule. This is assumed to be specific to the problem considered in this thesis, due to the nature of the cost function. Recall that the cost function consists of three components, namely product waste, underachievement and space violation, and that a small change in the replenishment order schedule will not necessarily result in any change of these three measures. It therefore frequently occurs that the cost function values of neighbouring solutions are the same, which is why the *EqualLimit* parameter was introduced in the simulated annealing algorithm in Chapter 5.

The most promising combination of values for the other three parameters were found during Run 18, namely *AcceptancesLimit* = $N = 2160$, *EqualLimit* = 25 and $\alpha = 0.9$. The cost function value associated with the best replenishment order schedule found during this search run is R 427 905.32. The overall best combination of parameter values are therefore the three aforementioned values combined with $\chi_0 = 0.2$ and a penalty value of 1. It is interesting to note that a few of the other combinations of parameter values yielded cost function values that are close to that of Run 18.

7.4.4 Additional results

It is undeniable that the so-called *best solutions* found during the 35 search runs actually correspond to very large cost function values. The cost function values are, however, slightly better than those associated with the initial solution and the solution employed by the retailer. Upon further inspection, it may be seen that the largest portions of the cost function values may be attributed to violations of the shelf space constraint. It is possible that the total amount of shelf space available is an inaccurate representation of the real-world resource, or perhaps even more

TABLE 7.4: Breakdown of the cost function values of the best solutions found during three additional simulated annealing search runs that were performed to evaluate the effect of the available shelf space.

Run	Available shelf space	Best solution found	Unmet demand	Waste	Space violation
36	200 000 cm^2	R 80 063	R 9 030	R 12 495	R 58 538
37	300 000 cm^2	R 27 990	R 8 874	R 19 117	—
38	500 000 cm^2	R 23 322	R 7 317	R 16 005	—

possible that the unit space values of the individual products are slightly incorrect.

Additional search runs were therefore performed to investigate the effect of increasing the available shelf space. The daily shelf space utilisation values associated with the retailer's replenishment order schedule were calculated according to (3.2). These utilisation values were used as guidelines to choose values for the total amount of available shelf during additional search runs. The minimum daily shelf space utilisation associated with the retailer's replenishment order schedule is 116 091.5 cm^2 , the maximum daily shelf space utilisation is 249 393.7 cm^2 and the average daily shelf space utilisation is 169 954.8 cm^2 . It was therefore decided that the first additional search run should be based on 200 000 cm^2 available shelf space, which might be slightly too small for the retailer's replenishment order schedule, while the second additional search run was performed with a sufficient 300 000 cm^2 available shelf space and the final additional search run correspond to an ample shelf space of 500 000 cm^2 .

The results obtained from these three additional search runs are shown in Table 7.4. Monetary values are rounded to the nearest Rand in the table. The cost function values associated with the best solutions found are far more promising than before. A breakdown of the cost function values is also provided in Table 7.4. Recall that the cost function consists of three components. The first component is as a result of unmet demand, which is the underachievement associated with having insufficient product units on display to satisfy the forecasted demand and provide for safety stock. The second component is as a result of product waste, which reflects product units that are discarded after reaching the end of their shelf lives. The final component is the penalty that accounts for daily violations of the available shelf space.

It is interesting to note that there exist unmet demand in all three cases, although there is sufficient shelf space available (or violation of the available shelf space, as in Run 36). This may possibly be attributed to the fact that it is more profitable not to order another pack of a specific product and suffer the loss of unmet demand than to order the additional pack and discard most of it as waste. Another possible explanation of the presence of unmet demand is that it is a result of the manner in which a new neighbouring solution is generated in the simulated annealing algorithm — by choosing two order quantities of a single product at random, decreasing the one order quantity by one and increasing the other order quantity by one, which may result in one or two undesirably small order quantities. In each of the three replenishment order schedules, there does, however, exist waste, as shown in Table 7.4. It is fair to assume that there will always be some waste because of the fixed pack sizes, especially when as many as 72 products are involved. But when the waste for the decision period is, for example, R 19 117, it translates to a daily waste value of approximately R 9.00 per product, which is a relatively small amount. Furthermore, the waste calculation is theoretical and depends on the theoretical shelf life value of a product, which means that, in real life, retailers may choose to still sell product units that are past their expiry dates, especially in a low-income environment. Finally, space violation only exists during Run 36, which is an expected result, since it was assumed that Run 36 is the only run where the available shelf space is insufficient, based on the shelf space utilisation of the retailer's replenishment order schedule.

TABLE 7.5: Breakdown of the cost function values associated with the retailer's replenishment order schedule when different amounts of shelf space are available.

No	Available shelf space	Cost function value	Unmet demand	Waste	Space violation
1	120 213 cm^2	R 1 789 284	R 33 944	R 20 204	R 1 735 136
2	200 000 cm^2	R 230 456	R 33 944	R 20 204	R 176 308
3	300 000 cm^2	R 54 148	R 33 944	R 20 204	—
4	500 000 cm^2	R 54 148	R 33 944	R 20 204	—

7.5 Discussion

In order to compare the results achieved by the application of the simulated annealing algorithm embedded in the DSS of Chapter 6 with the replenishment order schedule actually employed by the retail outlet in Grassy Park, the cost function value associated with the retailer's replenishment order schedule may be calculated for the additional values of available shelf space and presented in the same format as Table 7.4. The breakdown of the cost function values of the four cases (the measured amount of shelf space, as well as the three increased amounts) is shown in Table 7.5. The values are once again rounded to the nearest Rand. The cost function value associated with 120 213 cm^2 available shelf space has been presented earlier, and the respective contributions of unmet demand, product waste and space violation may now be seen in Table 7.5. For each of the four cases considered, the contributions of unmet demand and waste are the same, since the order quantities remain unchanged.

The replenishment order schedule associated with the best combination of parameter values, which was found to be the combination of Run 18, is shown in Table 7.6. The cost function value associated with this replenishment order schedule is R 428 367.60, and a breakdown of the value is as follows. R 21 821.21 may be attributed to unmet demand, R 11 958.55 corresponds to products discarded as waste, while the largest contributor to the cost function value is the space violation penalty of R 394 587.80. It is interesting to note that, during Run 18, which yielded the best combination of parameter values, no reheating occurred. This may be an indication that reheating was implemented too early during some of the other search runs, especially those corresponding to low-quality solutions.

When critically analysing the replenishment order schedule shown in Table 7.6, it is found that the schedule exhibits unusually large order quantities towards the end of the decision period. For example, the order quantity for Product 10 on Day 30 is 14. This may be attributed to the fact that the order quantities of the last few days are not always accounted for in some of the components of the cost function. Therefore, when the simulated annealing move operator affects changes resulting in an increased order quantity on one of the last days of the decision period, it does not have an effect on the cost function value and is not considered a non-improving solution.

This phenomenon may be explained as follows. First, a product may only contribute to the violation of the available shelf space once the product units arrives at the retail outlet, which is stipulated by its lead time. In this case study, large order quantities during the last two or three days of the decision period are therefore not penalised for causing a violation of the available shelf space during the first few days of the following period. Secondly, product units that are ordered towards the end of the decision period are not considered in the product waste calculation. Recall that product units are discarded as waste as soon as they reach the end of their shelf lives. Product units ordered during the last few days of the decision period, corresponding to their shelf lives, will only expire during the following decision period. The third component of

TABLE 7.6: *The replenishment order schedule associated with the best combination of simulated annealing parameter values (those of Run 18).*

Product	Day																															
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30		
1	0	0	0	3	0	3	2	1	0	1	2	1	3	0	0	0	0	2	1	0	1	1	1	1	2	3	1	3	0	0		
2	0	0	0	0	0	0	0	1	0	0	1	0	0	1	0	1	1	0	1	0	0	0	1	1	1	0	0	1	0	1		
3	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	2	0	0	0	0	0		
4	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	1	0	2	
5	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	1	0	1	0	0	1	0	0	0	1	0	4		
6	0	0	0	0	0	1	0	1	1	2	0	1	0	1	1	1	1	1	1	1	1	3	1	1	2	1	1	0	1	1	1	
7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	3	
9	0	0	0	0	1	0	1	0	0	0	1	0	2	0	0	0	0	0	1	0	1	0	0	1	0	0	1	0	0	1	0	1
10	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	14		
11	0	0	0	1	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	1	2	0	0	0	1	
12	0	0	0	0	0	2	0	1	1	1	1	3	0	2	0	2	1	1	2	0	2	1	1	2	0	1	3	1	1	3		
13	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	1	0	1	0	0	0	1	0	1	0	0	1	0	0	4		
14	0	0	0	1	0	1	1	0	0	2	0	0	1	0	0	1	1	0	1	0	0	1	0	1	0	1	0	0	1	1		
15	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	1	0	0	0	1	0	0		
16	0	0	0	0	1	0	0	0	0	1	1	0	0	0	1	0	0	1	0	1	0	0	0	0	0	1	1	1	0	0	3	
17	0	0	0	0	1	0	0	1	0	1	1	0	1	0	1	0	0	1	1	0	0	1	0	2	0	0	1	0	0	3		
18	0	0	0	0	1	0	1	0	0	0	2	0	0	0	0	2	0	0	1	0	0	2	1	0	0	0	1	1	2	3		
19	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0		
20	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0		
21	0	0	0	0	2	0	0	1	0	0	1	1	1	0	1	1	1	1	0	0	0	0	0	1	0	0	1	0	0	4		
22	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
23	0	0	0	1	3	1	5	5	0	4	1	3	2	3	2	1	4	1	2	3	3	2	2	3	1	2	3	1	2	17		
24	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	1	0	1	0	0		

TABLE 7.6: (continued) The replenishment order schedule associated with the best combination of simulated annealing parameter values (those of Run 18).

Product	Day																														
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	
25	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	
26	0	0	0	0	0	0	0	1	0	1	1	1	0	1	0	1	0	0	1	0	0	0	0	1	0	1	0	0	2	0	
27	0	0	0	4	4	4	1	2	3	3	2	3	5	1	0	2	3	3	1	4	0	1	3	2	2	3	1	2	2	6	
28	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	
29	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	
30	0	0	0	0	4	0	1	0	2	1	0	2	0	1	3	0	1	1	0	2	0	1	1	1	1	1	1	1	0	5	
31	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	
32	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	
33	0	0	0	0	0	0	0	1	0	1	1	0	1	0	2	0	0	1	0	1	1	0	0	1	1	1	0	1	0	3	
34	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	
35	0	0	0	0	0	0	2	0	0	1	0	1	0	0	1	0	0	1	1	0	0	2	1	1	1	0	1	1	2	1	
36	0	0	0	1	1	1	0	3	1	0	1	0	0	0	1	1	0	1	0	1	1	1	0	1	0	2	1	0	2	1	
37	0	0	0	5	4	2	3	2	4	5	5	0	5	10	2	0	2	2	3	0	4	2	6	12	2	0	10	0	4	7	
38	0	0	0	2	5	2	1	2	2	2	2	3	2	0	3	1	1	3	0	1	2	1	3	1	1	2	1	4	4		
39	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	
40	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
41	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	1	0	0	1	0	0	3	
42	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	2	
43	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	
44	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	
45	1	0	1	1	1	0	1	1	0	2	1	1	0	1	1	1	0	0	1	1	0	1	0	1	5	1	2	0	3	2	
46	0	0	0	0	1	3	9	6	5	5	3	3	5	2	5	0	7	1	3	1	8	1	2	6	2	3	4	1	1	11	
47	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	
48	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	2	1	0	0	

TABLE 7.6: (continued) The replenishment order schedule associated with the best combination of simulated annealing parameter values (those of Run 18).

Product	Day																														
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	
49	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	
50	7	2	2	5	2	0	8	4	3	9	0	12	0	2	2	7	1	1	5	8	1	8	4	4	13	1	3	3	5	14	
51	0	0	0	0	0	1	0	0	0	1	1	0	0	0	1	0	0	0	0	0	1	1	0	1	0	0	0	0	0	3	
52	0	0	0	0	1	0	0	0	1	1	0	0	1	1	0	0	1	0	0	1	0	1	0	0	1	0	0	1	0	4	
53	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	1	
54	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	1	0	1	0	0	0	0	0	0	1	0	0	
55	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	1	2	
56	0	0	0	0	0	0	1	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	
57	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	
58	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	
59	0	0	0	0	0	0	1	0	1	1	1	0	1	1	0	1	0	0	0	1	2	1	1	0	0	0	1	1	1	2	
60	0	0	0	1	2	0	1	1	1	1	1	1	0	2	1	1	0	3	1	0	0	2	2	0	0	3	0	1	1	4	
61	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	
62	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	1	0	1	0	
63	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	2
64	0	0	0	0	0	0	1	0	1	0	1	1	0	0	1	0	2	1	0	1	1	1	1	0	0	1	2	0	1	0	
65	0	0	0	0	0	1	0	0	0	1	0	0	0	1	0	0	1	0	0	0	0	1	0	0	0	0	0	1	0	1	
66	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	1	2	
67	0	0	0	0	0	0	1	0	0	0	1	0	0	1	0	0	0	1	1	0	0	1	0	0	1	0	1	1	0	0	
68	0	0	0	0	0	0	1	0	0	0	1	0	0	0	1	0	0	1	0	0	0	0	0	0	1	0	0	0	1	0	
69	0	0	0	0	1	0	0	0	1	1	0	1	1	0	0	2	0	0	1	0	0	1	0	1	0	1	0	0	1	2	
70	0	0	0	0	0	1	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	1	0	1	0	0	0	0	0	1	
71	0	0	0	0	0	1	1	0	1	1	0	0	0	0	1	1	0	1	0	1	0	1	0	1	0	1	0	0	0	2	
72	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	1	

the cost function value, namely the underachievement of displaying sufficient units to satisfy the demand and to maintain a suitable level of safety stock, is not influenced by the large order quantities during the last few days of the decision period. Recall that forecasted demand for the first three days of the following decision period were included in the input data in order to generate a replenishment order schedule for the current decision period, accounting for products' lead times.

A potential solution to this anomaly would be to solve the problem over a longer period than the actual decision period. Alternatively, a replenishment order schedule may be recomputed more frequently. For example, the replenishment order schedule in Table 7.6 might be implemented for the first 15 days of the decision period, after which the problem may again be solved for the next period of 30 days, but only implemented for the first 15 days of that period. It might be sensible to base the length of the additional period on the maximum shelf life length among the products, since the calculation of a product's waste is significantly influenced by its shelf life. Since a product's lead time is generally shorter than its shelf life, an additional period corresponding to the length of its shelf life should also prevent order quantities that induce shelf space violations.

In comparison to the replenishment order schedule implemented by the retailer, which is included in Appendix D, it is furthermore interesting to note that there is much variability in the order quantities of the product stipulated in Table 7.6, especially among the products for which large numbers of packs are generally ordered. The replenishment order schedule implemented by the retailer involves orders of sizes which are more consistent. It is not clear what level of consistency is preferred. An additional objective may be incorporated into the model of Chapter 3 which measures the layout quality of products in the fresh produce department associated with a replenishment order schedule. Consistency among order sizes is expected to be important in the presence of such an additional model objective.

In order to fully compare the best solutions found by adopting the best parameter value combination for the simulated annealing algorithm with the replenishment order schedule that was actually employed by the retailer in Grassy Park, the cost function values associated with the four different values of available shelf space for both the simulated annealing algorithm and the retailer's replenishment order schedule are shown in Table 7.7. The replenishment order schedules produced by the simulated annealing algorithm clearly outperform the retailer's replenishment order schedule in all four cases by a large margin.

This result may serve as verification that the simulated annealing algorithm of Chapter 5 is able to solve the model of Chapter 3. It is, however, important that these results are interpreted as being based on the assumptions and objectives that were deemed relevant when the mathematical modelling approach was described in Chapter 3. The replenishment order schedule that was actually employed by the retailer may be the better choice in the light of other objectives and assumptions. For example, another objective may be to maximise the overall freshness of products, or it may be assumed that product waste is calculated according to an alternative method.

7.6 Chapter summary

A real-life case study was performed in this chapter. The problem instance considered in this case study involves the ambient section of the fresh produce department of a retail outlet in Grassy Park, Cape Town, containing a total of 72 products. The problem instance was solved over a decision period of one month.

TABLE 7.7: Comparison of the cost function values associated with the best solutions found when adopting the best parameter value combination in the simulated annealing algorithm and the cost function values associated with the replenishment order schedule implemented by the retailer, for four values of available shelf space.

Available shelf space	Simulated annealing algorithm	Retailer's replenishment order schedule	Improvement
120 213 cm^2	R 428 368	R 1 789 284	76.1%
200 000 cm^2	R 80 063	R 230 456	65.3%
300 000 cm^2	R 27 990	R 54 148	48.3%
500 000 cm^2	R 23 322	R 54 148	56.9%

As part of the case study, a parameter evaluation experiment was performed in respect of five influential parameters incorporated in the simulated annealing algorithm. Several simulated annealing search runs with systematically varied combinations of parameter values were executed in order to find a suitable combination of these parameters that yields good results.

Upon analysis of the results thus obtained, it was found that the input data may contain inaccuracies in respect of the real-world situation. The results were nevertheless deemed satisfactory, since the input data were kept constant throughout the experiment and during the ensuing comparisons. Additional search runs with varied amounts of available shelf space were performed to conclude the investigation. The results were finally compared to a replenishment order schedule that was actually implemented in the retail outlet over the same time period and it was found that the replenishment order schedules recommended by the DSS of Chapter 6 yielded substantially superior results.

CHAPTER 8

Conclusion

Contents

8.1 Thesis summary	107
8.2 Appraisal of contributions	109
8.3 Suggestions for future work	111

The purpose of this chapter is to provide the reader with a brief overview of the work reported in this thesis. A summary of the thesis contents is provided in §8.1, with specific reference to the fulfilment of the thesis objectives identified in Chapter 1. An appraisal of the contributions of the thesis follows in §8.2, incorporating feedback obtained from a representative of the retail sector. Suggestions for future work are finally made in §8.3, in fulfilment of Thesis Objective VII of §1.3.

8.1 Thesis summary

An introduction to the problem considered in this thesis was given in Chapter 1. Background information was first provided to the problem in §1.1, and this was followed by an outline of the research project aim and scope in §1.2. Seven objectives were identified for pursuit in this thesis and these objectives were described in §1.3 before an overview of the organisation of material in the thesis was provided in §1.4.

A review of the literature on relevant topics identified in Thesis Objective I was conducted in Chapter 2. The first topic of the review, in §2.1, was the retail industry in general, with a focus on basic retail definitions and descriptions of general retail operations, customer behaviour in a retail outlet and the nature of the South African retail industry. In respect of fresh produce, the product category of perishable products and the topic of food waste were considered in §2.2. A brief review followed in §2.3 of fields of study that are related to the product replenishment order decisions of retailers, namely product assortment, price planning, inventory management and shelf space allocation. Since the mathematical model proposed in this thesis takes the form of a combinatorial optimisation model, a brief overview of solution techniques for combinatorial optimisation problems was provided in §2.4. Techniques that may be employed to validate mathematical models were described in §2.5, while general requirements and essential considerations pertaining to the design and development of DSSs were discussed in §2.6. The literature review stands in fulfilment of Thesis Objective II.

Chapter 3 was devoted to the formulation of a novel mathematical model in support of a retailer's fresh produce replenishment order decisions, in pursuit of Thesis Objective III. The formulation of the model was documented, starting with the underlying assumptions and various considerations which influence the mathematical model, in §3.1. The objective function, which may be interpreted as a lost opportunity cost, was discussed in §3.2. The first model constraint, associated with the limited amount of shelf space available, was introduced in §3.3. A constraint set for ensuring expected demand satisfaction was put forward in §3.4, and this was followed by an explanation of the method followed to model conservation of stock in §3.5. A product waste constraint, together with an accompanying example, was included in §3.6. The model was finally presented in its entirety in §3.7 in the form of a chapter summary.

In pursuit of Thesis Objective IV, two primary solution approaches were developed for solving the mathematical model of Chapter 3. The first of these approaches was presented in Chapter 4. It is an exact (branch-and-cut) solution methodology implemented in IBM ILOG CPLEX Optimization Studio (CPLEX). The required input data and computer implementation details of this solution approach were discussed in §4.1. Several small problem instances were solved in order to demonstrate the functioning of the approach, using hypothetical problem data proposed in §4.2, and the solutions to these problem instances were analysed in §4.3. In an attempt to find a more effective solution approach in CPLEX, a solution approximation method and a problem-specific heuristic for solving the model of Chapter 3 were furthermore presented in §4.4, before the chapter closed with a summary in §4.5.

An approximate solution approach, based on the method of simulated annealing, was presented in Chapter 5. The building blocks of the simulated annealing algorithm incorporated in the solution approach were discussed in §5.1, namely the constraint handling technique employed, the method of initial solution construction, the simulated annealing move operator implemented, the acceptance rule for neighbouring solutions adopted, the method of initial temperature calculation, the cooling and reheating schedules implemented and, finally, the search termination criteria imposed. A pseudocode description of the algorithm and various computer implementation details were also provided. In §5.2, the simulated annealing algorithm implementation was validated by solving several hypothetical problem instances using both the exact and approximate solution approaches. The operation of the simulated annealing algorithm was demonstrated in §5.3 by solving a larger hypothetical problem instance, and this was followed by a brief chapter summary in §5.4.

A DSS for fresh produce replenishment orders was designed in Chapter 6 in fulfilment of Thesis Objective V. The work of the previous chapters was brought together in the design process to produce a support tool for fresh produce replenishment order decisions. The architecture of the DSS was first discussed in §6.1 and specific reference was made to the three main components of any DSS, namely a database, a UI and a model base. In §6.2, the functioning of the DSS was elucidated by means of a walk-through description of a concept demonstrator UI for the DSS. A chapter summary was finally provided in §6.3.

A realistic case study was finally performed in Chapter 7, in pursuit of Thesis Objective VI. The chapter opened in §7.1 with background information on the case study, which involved 72 products in the ambient section of the fresh produce department within a retail outlet in Grassy Park, Cape Town. In §7.2, the case study data and the necessary data preparation process were described. An algorithmic parameter evaluation experiment was designed in §7.3 to find a set of suitable values for selected parameters in the simulated annealing algorithm when solving the mathematical model in the context of the case study problem. The results of the experiment were discussed in §7.4. The parameter values thus uncovered were incorporated in the solution approach in order to recommend a fresh produce replenishment order schedule for a one-month

period, which was compared in §7.5 with a schedule that was actually employed at the retail outlet over the same time period. After a discussion on the noticeable differences and similarities between the two replenishment order schedules, a chapter summary was included in §7.6.

8.2 Appraisal of contributions

This section contains an appraisal of the thesis contributions and consists of two parts. An appraisal of the modelling framework, which refers to the mathematical model proposed in Chapter 3 and the DSS of Chapter 6 in which this model was embedded, is provided in §8.2.1. An appraisal of the case study, including feedback from the relevant industry partner, is subsequently provided in §8.2.2.

8.2.1 Appraisal of the modelling framework

The research performed in this thesis was initiated by a problem that exists in the case study retail outlet. No sophisticated methods are used in the determination of replenishment order scheduling for fresh produce in this retail outlet. By merely considering the fresh produce display in this outlet, however, it was possible to ascertain that too many units of specific products are on display, detracting from the image of freshness and quality portrayed by the remainder of the fresh produce department.

The case study retail outlet is characterised as a special type of retail outlet, namely one with no backroom storage area for fresh produce. The mathematical model proposed in Chapter 3 represents several aspects of the operations at this type of retail outlet, such as satisfying customer demand, maintaining a level of safety stock, maximising profit, minimising waste and accounting for the fact that a limited amount of shelf space is available.

From the literature review of Chapter 2 it is clear that product waste is a serious concern and deserves attention when making fresh produce replenishment order decisions. Waste is calculated theoretically in the mathematical model in a way that makes practical sense, as illustrated by the example in §3.6. After some experimentation it was, however, found that the maximum operator in (3.11) is required in the model constraint associated with product waste, since waste values of consecutive days initially cancelled out during the calculation of the objective function value and low-quality solutions were therefore incorrectly seen as high-quality solutions.

Retailers may nevertheless opt to continue the display of fresh produce that have already reached the end of their shelf lives. This type of behaviour would influence replenishment order decisions, but is difficult to incorporate in a mathematical model. The fact that future product waste is projected as part of the model outcome is considered a useful contribution, as pointed out by the industry partner [153]:

“The model outcome allows the user visibility of possible waste during the following ordering period. In practice this will allow the retailer to manage the mark-down of products more cost effectively. Currently, products are marked down to about 50% of the retail price the afternoon prior to expiry. If the retailer has visibility of products that might be expiring within the next 3–5 days, a gradual mark-down strategy can be put in place to maximise profit. This function will be widely accepted by retailers.”

There are several differences between the two solution approaches implemented to solve the

mathematical model of Chapter 3. In the exact solution approach, the objective function and the constraints could be implemented directly in CPLEX and provision was made in the CPLEX programming code to incorporate input data. In the approximate solution approach, on the other hand, model constraints other than the shelf space limitation were interpreted as mere calculations required to obtain the objective function value. Furthermore, a penalisation associated with violating the shelf space constraint was incorporated in the approximate solution approach.

The approximate solution technique of simulated annealing was deemed a desirable approach, since one of the model inputs is forecasted demand. If the demand values were exact, more emphasis on finding an exact solution to the model may have been warranted. Moreover, it is highly probable that errors will occur in the ordering process at the distribution centre, resulting in additional packs being delivered at a retail outlet. Improvisation in the utilisation of the available shelf space would then be required. Fortunately, floor space reserved for the movement of customers may be utilised within the case study outlet in such a case, although it would not be ideal to limit customer movement too much.

A flexible and user-friendly DSS concept demonstrator was implemented in this thesis. The DSS allows a user to change the input data related to the fresh produce department of the retail outlet under consideration when calculating a replenishment order schedule. If the user is interested, (s)he may also make changes to the parameters incorporated in the simulated annealing solution approach. This is, however, not a necessity, since suitable default parameter values are provided. The industry partner elaborated on the flexibility of the system in respect of retailer decisions [153]:

“Due to the functionality of the model allowing for multiple variables and the versatility thereof, the model can be used to experiment with the required increase in shelf space that correlates with future demand increase. Based on customer growth projections, this will allow the retailer to plan and budget for any possible expenses relating to such expansions.”

8.2.2 Appraisal of the case study

During the case study, it was found that the cost function value associated with the recommended replenishment order schedule is substantially lower than the cost function value associated with the replenishment order schedule that was actually employed at the Grassy Park retail outlet during the same time period. On the basis of the factors accounted for in the cost function, it may therefore be concluded that the proposed approximate solution approach returns satisfactory solutions to instances of the mathematical model. It was furthermore found that the amount of shelf space available in the ambient section of the Grassy Park retail outlet may be insufficient, since violation of the shelf space constraint contributed significantly in the calculation of the cost function value associated with the replenishment order schedule actually implemented at the Grassy Park retail outlet. The industry partner provided the following feedback in this respect [153]:

“The model outcome indicates that the current 57 crates used in the store do not allow enough display area to satisfy customer demand. Although the retailer’s customer demand fulfilment is within the desired parameters, it will be valuable to investigate this aspect as part of the continuous improvement strategy. If it is determined that an increase in shelf space is financially viable, the retailer would most definitely consider a redesign of the current store layout to allow for more fresh produce display.”

As explained in §7.3, the cost function values associated with replenishment order schedules are influenced by the input data. If more accurate input data could be obtained, especially in respect of the unit space taken up by the respective products in the fresh produce department, the results would be a better representation of the real-world situation. The input data were, however, kept constant during comparisons and the recommended and actual replenishment order schedules are therefore compared on a fair basis.

In practice, retailers are also concerned with other aspects of operations at a retail outlet. The total number of packs ordered during a decision period is, for example, a measurement that retailers may be interested in, which is an alternative basis of comparison not considered in the case study of Chapter 7. Furthermore, retailers may also aim to maintain an optimal level of safety stock. In the modelling framework of Chapter 3, the safety stock values were assumed to be fixed product characteristics calculated from a product's average daily demand and lead time, prior to the application of the mathematical model. During final discussions with the representative of the retail industry, it was realised that an alternative formula for calculating the safety stock values of fresh produce may be more profitable to a retailer. The safety stock formula currently employed in the modelling framework is more suitable to products with longer shelf lives.

8.3 Suggestions for future work

The suggestions as to possible future work documented in this section are primarily related to the mathematical model proposed in Chapter 3. Several draft mathematical models were considered before the model proposed in Chapter 3 was finally settled upon. Since the model is a simplification of the operations at the type of retail outlet under consideration, there are many additional factors that may be taken into account in the model, and aspects that are currently incorporated in the model may also be modelled in alternative ways.

Alternative methods of incorporating the notion of safety stock may, for example, be investigated. As mentioned above, a formula tailored specifically for fresh produce may yield more accurate safety stock values. In the mathematical model of Chapter 3, safety stock was considered together with the daily forecasted customer demand, so as to consider both these aspects simultaneously in replenishment decisions. Separating the safety stock replenishment decisions from demand replenishment decisions may, however, be an interesting investigation. The industry partner provided an explanation as to why it is not necessary for the calculation of safety stock values for fresh produce to take the full lead time into account [153]:

“The current safety stock calculation does not take into consideration that only the very first ordering instance is subjected to the full lead time. Products, especially fresh produce, are delivered multiple times per week, thus eliminating the effect of the lead time. For example, if one delivery does not happen as planned and produce is delivered 6 days of a week instead of 7, there will only be one day in the week that the store will not receive products, which would be covered by the safety stock.”

As mentioned in §7.5, an additional objective may also be considered in the mathematical model. Currently, the sum of the underachievement associated with having sufficient products on display and products discarded as waste during a decision period is minimised. As a result, two replenishment order schedules that look very different may yield the same cost function value, although one of the two schedules may be preferred above the other in real life. An alternative approach would be to include the objective of maximising overall freshness of products in the

fresh produce department over a specified decision period. Furthermore, instead of only taking individual product profits into account in the objective function that is to be minimised, the objective function may be changed to explicitly maximise the overall profit generated during a decision period in the fresh produce department of a retail outlet.

The outcome of the model furthermore often stipulates that no packs of a specific product should be ordered during a decision period. The reason for this is that the number of product units that will be discarded as waste when ordering the product in an attempt to satisfy the forecasted demand far outweighs the loss associated with not satisfying customer demand, usually as a result of large pack sizes in combination with low demand values for the product and a short product shelf life. The industry partner highlighted the possibility of pursuing a trade-off between a reduction in production waste and the negative effects of a smaller product assortment as future work [153]:

“The model calculates that some products should not be included in the ordering cycle based on the demand forecast in comparison to the possibility of product waste. Although the negative effect of a reduction in product assortment will have to be investigated and analysed financially, this is an interesting view that should be investigated in future. If such trends persist, the retailer might adjust the product assortment accordingly.”

An experiment may furthermore be conducted to compare different move operators in the simulated annealing algorithm. The move operator has a significant influence on the direction of the simulated annealing search process. Moreover, an additional solution approach may be implemented to solve the mathematical model. A problem-specific heuristic in which the specific aims of a retailer guide the search for satisfactory solutions may, for example, yield acceptable results in a shorter time period than currently achieved by the simulated annealing algorithm.

The requirements for the model’s input data may also be changed. In general, more accurate and reliable data would produce more reliable output results. Since different retailers probably have different sets of data available, the model may be changed to take the specific available data into account. For example, several models in the literature take shelf space elasticity into account (*i.e.* the effect of the amount of shelf space allocated to a product on the demand for that product). If a retailer collects such data, and it could be incorporated in the mathematical model, it would certainly enhance the validity of the model and the quality of its solutions.

Finally, the DSS may be improved further so as to focus more on the requirements of employees who will use such a system to make replenishment order decisions. Interviews may be conducted with potential users and DSS prototypes may be compared in order to determine the considerations that these users regard as important, thereby highlighting additional functionalities which they desire in such a system.

References

- [1] AGRAWAL N & SMITH S, 2009, *Retail supply chain management: Quantitative models and empirical studies*, Springer, New York (NY).
- [2] AGRAWAL N & SMITH SA, 2013, *Optimal inventory management for a retail chain with diverse store demands*, European Journal of Operational Research, **225(3)**, pp. 393–403.
- [3] AIELLO G, LA SCALIA G & MICALE R, 2012, *Simulation analysis of cold chain performance based on time-temperature data*, Production Planning and Control: The Management of Operations, **23(6)**, pp. 468–476.
- [4] AMROUCHE N & ZACCOUR G, 2007, *Shelf-space allocation of national and private brands*, European Journal of Operational Research, **180(2)**, pp. 648–663.
- [5] ANDERSON EE & AMATO HN, 1974, *A mathematical model for simultaneously determining the optimal brand-collection and display-area allocation*, Operations Research, **22(1)**, pp. 13–21.
- [6] ANDERSSON H, HOFF A, CHRISTIANSEN M, HASLE G & LOKKETANGEN A, 2010, *Industrial aspects and literature survey: Combined inventory management and routing*, Computers and Operations Research, **37(9)**, pp. 1515–1536.
- [7] ARNOTT D & PERVAN G, 2008, *Eight key issues for the decision support systems discipline*, Decision Support Systems, **44(3)**, pp. 657–672.
- [8] AUNG MM & CHANG YS, 2014, *Temperature management for the quality assurance of a perishable food supply chain*, Food Control, **40**, pp. 198–207.
- [9] BAI R, BURKE EK & KENDALL G, 2008, *Heuristic, meta-heuristic and hyper-heuristic approaches for fresh produce inventory control and shelf space allocation*, Journal of the Operational Research Society, **59**, pp. 1387–1397.
- [10] BAI R, 2005, *An investigation of novel approaches for optimising retail shelf space allocation*, PhD Thesis, University of Nottingham, Nottingham.
- [11] BAI R & KENDALL G, 2008, *A model for fresh produce shelf-space allocation and inventory management with freshness-condition-dependent demand*, INFORMS Journal on Computing, **20(1)**, pp. 78–85.
- [12] BAI R, VAN WOENSEL T, KENDALL G & BURKE EK, 2013, *A new model and a hyper-heuristic approach for two-dimensional shelf space allocation*, 4OR, **11**, pp. 31–55.
- [13] BAKKER M, RIEZEBOS J & TEUNTER RH, 2012, *Review of inventory systems with deterioration since 2001*, European Journal of Operational Research, **221(2)**, pp. 275–284.
- [14] BARON O, BERMAN O & PERRY D, 2011, *Shelf space management when demand depends on the inventory level*, Production and Operations Management, **20(5)**, pp. 714–726.

- [15] BARRY CS & GIOVANNONI JJ, 2007, *Ethylene and fruit ripening*, Journal of Plant Growth Regulation, **26**, pp. 143–159.
- [16] BEAMER BG & PRESTON WP, 1991, *Shelf space allocation in the produce department: Implications for marketing specialty produce*, Journal of Food Distribution Research, **22(3)**, pp. 23–36.
- [17] BENEKE J, 2010, *Consumer perceptions of private label brands within the retail grocery sector of South Africa*, African Journal of Business Management, **4(2)**, pp. 203–220.
- [18] BITRAN GR & MONDSCHEN SV, 1997, *Periodic pricing of seasonal products in retailing*, Management Science, **43(1)**, pp. 64–79.
- [19] BIXBY RE, 2002, *Solving real-world linear programs: A decade and more of progress*, Operations Research, **50**, pp. 3–15.
- [20] BOOKBINDER JH & ZAROOR FH, 2001, *Direct product profitability and retail shelf-space allocation models*, Journal of Business Logistics, **22(2)**, pp. 183–208.
- [21] BORIN N, FARRIS PW & FREELAND JR, 1994, *A model for determining retail product category assortment and shelf space allocation*, Decision Sciences, **25(3)**, pp. 359–384.
- [22] BOYD DE & BAHN KD, 2009, *When do large product assortments benefit consumers? An information-processing perspective*, Journal of Retailing, **85(3)**, pp. 288–297.
- [23] BRIESCH RA, CHINTAGUNTA PK & FOX EJ, 2009, *How does assortment affect grocery store choice?*, Journal of Marketing Research, **46(2)**, pp. 176–189.
- [24] BROWN WM & TUCKER WT, 1961, *Vanishing shelf space*, Atlanta Economic Review, **9**, pp. 9–13.
- [25] BULTEZ A & NAERT P, 1988, *SH.A.R.P.: Shelf allocation for retailers' profit*, Marketing Science, **7(3)**, pp. 211–231.
- [26] BURNS DJ & NEISNER L, 2006, *Customer satisfaction in a retail setting: The contribution of emotion*, International Journal of Retail and Distribution Management, **34(1)**, pp. 49–66.
- [27] BUZBY J, BENTLEY J, PADERA B, AMMON C & CAMPUZANO J, 2015, *Estimated fresh produce shrink and food loss in U.S. supermarkets*, Agriculture, **5**, pp. 626–648.
- [28] CACHON G, 2001, *Managing a retailer's shelf space, inventory, and transportation*, Manufacturing and Service Operations Management, **3(3)**, pp. 211–229.
- [29] CACHON GP & FISHER M, 2000, *Supply chain inventory management and the value of shared information*, Management Science, **46(8)**, pp. 1032–1048.
- [30] CAMPO K & GIJSBRECHTS E, 2005, *Retail assortment, shelf and stockout management: Issues, interplay and future challenges*, Applied Stochastic Models in Business and Industry, **21(4-5)**, pp. 383–392.
- [31] CASTELLI M & VANNESCHI L, 2014, *Genetic algorithm with variable neighborhood search for the optimal allocation of goods in shop shelves*, Operations Research Letters, **42(5)**, pp. 355–360.
- [32] ČERNÝ V, 1985, *Thermodynamical approach to the traveling salesman problem: An efficient simulation algorithm*, Journal of Optimization Theory and Applications, **45(1)**, pp. 41–51.
- [33] CHEN M & LIN C, 2007, *A data mining approach to product assortment and shelf space allocation*, Expert Systems with Applications, **32(4)**, pp. 976–986.

- [34] CHRISAFIS A, 2016, *French law forbids food waste by supermarkets*, [Online], [Cited November 17th, 2016], Available from %5Curl%7Bhttps://www.theguardian.com/world/2016/feb/04/french-law-forbids-food-waste-by-supermarkets%7D.
- [35] CORSTJENS M & DOYLE P, 1981, *A model for optimizing retail space allocations*, *Management Science*, **27(7)**, pp. 822–833.
- [36] DE AGUIAR MT, 2015, *The retail shelf space allocation problem: New optimisation methods applied to a supermarket chain*, PhD Thesis, University of Porto, Porto.
- [37] DESMET P & RENAUDIN V, 1998, *Estimation of product category sales responsiveness to allocated shelf space*, *International Journal of Research in Marketing*, **15**, pp. 443–457.
- [38] DORIGO M, 1992, *Optimization, learning and natural algorithms*, PhD Thesis, Politecnico di Milano, Milan.
- [39] DOUCET J, 2009, *Components, purpose and function of information systems*, [Online], [Cited November 15th, 2016], Available from %5Curl%7Bhttps://jennadoucet.wordpress.com/2010/03/14/components-purpose-and-function-of-information-systems/%7D.
- [40] DRÉO J, PÉTROWSKI A & TAILLARD E, 2006, *Metaheuristics for hard optimization*, Springer, Berlin.
- [41] EGGLESE RW, 1990, *Simulated annealing: A tool for operational research*, *European Journal of Operational Research*, **46**, pp. 271–281.
- [42] EISEND M, 2014, *Shelf space elasticity: A meta-analysis*, *Journal of Retailing*, **90(2)**, pp. 168–181.
- [43] ENGINEERING TOOLBOX, 2016, *Fruits and vegetables — Optimal storage conditions*, [Online], [Cited November 8th, 2016], Available from %5Curl%7Bhttp://www.engineeringtoolbox.com/fruits-vegetables-storage-conditions-d.710.html%7D.
- [44] ERLINKOTTER D, 1990, *Ford Whitman Harris and the economic order quantity model*, *Operations Research*, **38(6)**, pp. 937–946.
- [45] EROGLU C, WILLIAMS BD & WALLER MA, 2013, *The backroom effect in retail operations*, *Production and Operations Management*, **22(4)**, pp. 915–923.
- [46] FADILUĞLU MM, KARASAN OE & PINAR M, 2010, *A model and case study for efficient shelf usage and assortment analysis*, *Annals of Operations Research*, **180**, pp. 105–124.
- [47] FERNIE J & GRANT DB, 2008, *On-shelf availability: The case of a UK grocery retailer*, *International Journal of Logistics Management*, **19(3)**, pp. 293–308.
- [48] FOSSO WAMBA S, LEFEBVRE LA, BENDAVID Y & LEFEBVRE É, 2008, *Exploring the impact of RFID technology and the EPC network on mobile B2B eCommerce: A case study in the retail industry*, *International Journal of Production Economics*, **112(2)**, pp. 614–629.
- [49] GAJJAR HK & ADIL GK, 2010, *A piecewise linearization for retail shelf space allocation problem and a local search heuristic*, *Annals of Operations Research*, **179**, pp. 149–167.
- [50] GAUTENG PROVINCIAL TREASURY, 2012, *The retail industry on the rise in South Africa*, (Unpublished) Technical Report, Pretoria.
- [51] GEISMAR HN, DAWANDE M, MURTHI BPS & SRISKANDARAJAH C, 2015, *Maximizing revenue through two-dimensional shelf-space allocation*, *Production and Operations Management*, pp. 1–16.

- [52] GEMAN S & GEMAN D, 1984, *Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images*, IEEE Transactions on Pattern Analysis and Machine Intelligence, **6(6)**, pp. 721–741.
- [53] GHOSH P, TRIPATHI V & KUMAR A, 2010, *Customer expectations of store attributes: A study of organized retail outlets in India*, Journal of Retail and Leisure Property, **9(1)**, pp. 75–87.
- [54] GLOVER F, 1989, *Tabu search — Part I*, ORSA Journal on Computing, **1(3)**, pp. 190–206.
- [55] GOYAL S & GIRI B, 2001, *Recent trends in modeling of deteriorating inventory*, European Journal of Operational Research, **134(1)**, pp. 1–16.
- [56] GÜRLER Ü & ÖZKAYA BY, 2008, *Analysis of the (s,S) policy for perishables with a random shelf life*, IIE Transactions, **40**, pp. 759–781.
- [57] GUSTAVSSON J, CEDERBERG C, SONESSON U, VAN OTTERDIJK R & MEYBECK A, 2011, *Global food losses and food waste: Extent, causes and prevention*, Food and Agriculture Organization of the United Nations, Rome.
- [58] HALL JM, KOPALLE PK & KRISHNA A, 2010, *Retailer dynamic pricing and ordering decisions: Category management versus brand-by-brand approaches*, Journal of Retailing, **86(2)**, pp. 172–183.
- [59] HANSEN JM, RAUT S & SWAMI S, 2010, *Retail shelf allocation: A comparative analysis of heuristic and meta-heuristic approaches*, Journal of Retailing, **86(1)**, pp. 94–105.
- [60] HANSEN P & HEINSBROEK H, 1979, *Product selection and space allocation in supermarkets*, European Journal of Operational Research, **3(6)**, pp. 474–484.
- [61] HARIGA MA, AL-AHMARI A & MOHAMED ARA, 2007, *A joint optimisation model for inventory replenishment, product assortment, shelf space and display area allocation decisions*, European Journal of Operational Research, **181(1)**, pp. 239–251.
- [62] HILLIER FS & LIEBERMAN GJ, 2010, *Introduction to operations research*, 9th Edition, McGraw-Hill, Singapore.
- [63] HOLLAND J, 1975, *Adaptation in natural and artificial systems*, University of Michigan Press, Ann Arbor (MI).
- [64] HONHON D, GAUR V & SESHADRI S, 2010, *Assortment planning and inventory decisions under stockout-based substitution*, Operations Research, **58(5)**, pp. 1364–1379.
- [65] HORNIK K, 2016, *Frequently asked questions on R*, [Online], [Cited November 20th, 2016], Available from %5Curl%7Bhttps://CRAN.R-project.org/doc/FAQ/R-FAQ.html%7D.
- [66] HÜBNER AH, 2011, *Retail category management: Decision support systems for assortment, shelf space, inventory and price planning*, 1st Edition, Springer, Heidelberg.
- [67] HÜBNER AH & KUHN H, 2012, *Retail category management: State-of-the-art review of quantitative research and software applications in assortment and shelf space management*, Omega, **40(2)**, pp. 199–209.
- [68] HÜBNER AH, KUHN H & STERNBECK MG, 2013, *Demand and supply chain planning in grocery retail: An operations planning framework*, International Journal of Retail and Distribution Management, **41(7)**, pp. 512–530.
- [69] HUTTER F, HOOS HH & LEYTON-BROWN K, 2010, *Automated configuration of mixed integer programming solvers*, Proceedings of the International Conference on Integration of Artificial Intelligence (AI) and Operations Research (OR) Techniques in Constraint Programming, Bologna, pp. 186–202.

- [70] HWANG H, CHOI B & LEE M, 2005, *A model for shelf space allocation and inventory control considering location and inventory level effects on demand*, International Journal of Production Economics, **97(2)**, pp. 185–195.
- [71] IBM CORPORATION, 2015, *CPLEX user's manual, Version 12, Release 6*, [Online], [Cited May 3rd, 2016], Available from %5Curl%7Bhttp://www.ibm.com/support/knowledgecenter/api/content/nl/en-us/SSSA5P_12.6.3/ilog.odms.studio.help/Optimization_Studio/topics/PLUGINS_ROOT/ilog.odms.studio.help/pdf/usrcplex.pdf%7D.
- [72] IBM CORPORATION, 2015, *Getting started with CPLEX, Version 12, Release 6*, [Online], [Cited May 3rd, 2016], Available from %5Curl%7Bhttp://www.ibm.com/support/knowledgecenter/api/content/nl/en-us/SSSA5P_12.6.3/ilog.odms.studio.help/Optimization_Studio/topics/PLUGINS_ROOT/ilog.odms.studio.help/pdf/gscplex.pdf%7D.
- [73] IHAKA R & GENTLEMAN R, 1996, *R: A language for data analysis and graphics*, Journal of Computational and Graphical Statistics, **5(3)**, pp. 299–314.
- [74] INGBER L, 1996, *Adaptive simulated annealing*, Control and Cybernetics, **25**, pp. 33–54.
- [75] INVESTOPEDIA STAFF, 2016, *The industry handbook: The retailing industry*, [Online], [Cited October 26th, 2016], Available from %5Curl%7Bhttp://www.investopedia.com/features/industryhandbook/retail.asp%7D.
- [76] IOL REPORTER, 2016, *Pick n Pay gains on new stores*, [Online], [Cited October 25th, 2016], Available from %5Curl%7Bhttp://www.iol.co.za/business/companies/pick-n-pay-gains-on-new-stores-2080899%7D.
- [77] IRION J, LU J, AL-KHAYYAL F & TSAO Y, 2011, *A hierarchical decomposition approach to retail shelf space management and assortment decisions*, Journal of the Operational Research Society, **62**, pp. 1861–1870.
- [78] IRION J, AL-KHAYYAL F & LU J, 2012, *A piecewise linearization framework for retail shelf space management models*, European Journal of Operational Research, **222(1)**, pp. 122–136.
- [79] JAJJA M, 2013, *Retail shelf space allocation analysis using system dynamics approach*, Journal of Quality and Technology Management, **IX(I)**, pp. 99–129.
- [80] JOSHI R, BANWET D, SHANKAR R & GANDHI J, 2012, *Performance improvement of cold chain in an emerging economy*, Production Planning and Control: The Management of Operations, **23**, pp. 817–836.
- [81] KAR S, BHUNIA AK & MAITI M, 2001, *Inventory of multi-deteriorating items sold from two shops under single management with constraints on space and investment*, Computers and Operations Research, **28**, pp. 1203–1221.
- [82] KARAKUL M & CHAN L, 2008, *Analytical and managerial implications of integrating product substitutability in the joint pricing and procurement problem*, European Journal of Operational Research, **190**, pp. 179–204.
- [83] KARMAKAR B & CHOUDHURY KD, 2010, *A review on inventory models for deteriorating items with shortages*, Assam University Journal of Science and Technology, **6(2)**, pp. 51–59.
- [84] KENDALL G, 2014, *The science that makes us spend more in supermarkets, and feel good while we do it*, [Online], [Cited November 17th, 2016], Available from %5Curl%7Bhttp://theconversation.com/the-science-that-makes-us-spend-more-in-supermarkets-and-feel-good-while-we-do-it-23857%7D.

- [85] KENNEDY J, 1995, *Particle swarm optimization*, Proceedings of the IEEE International Conference on Neural Networks, Perth, pp. 1942–1948.
- [86] KHOUJA M, 1999, *The single-period (news-vendor) problem: Literature review and suggestions for future research*, Omega, **27(5)**, pp. 537–553.
- [87] KIRKPATRICK S, GELATT CD & VECCHI MP, 1983, *Optimization by simulated annealing*, Science, **220(4598)**, pp. 671–680.
- [88] KLOTZ E & NEWMAN AM, 2013, *Practical guidelines for solving difficult mixed integer linear programs*, Surveys in Operations Research and Management Science, **18(1)**, pp. 18–32.
- [89] KÖK AG, FISHER ML & VAIDYANATHAN R, 2008, *Assortment planning: Review of literature and industry practice*, pp. 99–153 in AGRAWAL N & SMITH SA (EDS), *Retail supply chain management*, Springer, New York (NY).
- [90] LAL R & MATUTES C, 1994, *Retail pricing and advertising strategies*, Journal of Business, **67(3)**, pp. 345–370.
- [91] LANDA-SILVA D, MARIKAR F & LE K, 2009, *Heuristic approach for automated shelf space allocation*, Proceedings of the ACM symposium on Applied Computing, Honolulu, pp. 922–928.
- [92] LANDRY M, MALOUIN J & ORAL M, 1983, *Model validation in operations research*, European Journal of Operational Research, **14(3)**, pp. 207–220.
- [93] LAW AM, 2009, *How to build valid and credible simulation models*, Proceedings of the 2009 Winter Simulation Conference, Austin (TX), pp. 24–33.
- [94] LENSTRA JK, 1997, *Local search in combinatorial optimization*, Princeton University Press, Princeton (NJ).
- [95] LEVY M, GREWAL D, KOPALLE PK & HESS JD, 2004, *Emerging trends in retail pricing practice: Implications for research*, Journal of Retailing, **80**, pp. xiii–xxi.
- [96] LEWIS C & RIEMAN J, 1994, *Task-centered user interface design: A practical introduction*, [Online], [Cited November 15th, 2016], Available from <http://hcibib.org/tcuid/>.
- [97] LI R, LAN H & MAWHINNEY JR, 2010, *A review on deteriorating inventory study*, Journal of Service Science and Management, **3**, pp. 117–129.
- [98] LI Y, CHEANG B & LIM A, 2012, *Grocery perishables management*, Production and Operations Management, **21(3)**, pp. 504–517.
- [99] LIM A, RODRIGUES B & ZHANG X, 2004, *Metaheuristics with local search techniques for retail shelf-space optimization*, Management Science, **50(1)**, pp. 117–131.
- [100] LIMA R, 2010, *IBM ILOG CPLEX — What is inside of the box?*, Proceedings of the Enterprise Wide Optimization Seminar, Pittsburgh (PA).
- [101] LIN CSK, PFALTZGRA LA, HERRERO-DAVILA L, MUBOFU EB, ABDERRAHIM S, CLARK JH, KOUTINAS AA, KOPSAHELIS N, STAMATELATOU K, DICKSON F, THANKAPPAN S, MOHAMED Z, BROCKLESBY R & LUQUE R, 2013, *Food waste as a valuable resource for the production of chemicals, materials and fuels*, Energy and Environmental Science, **6**, pp. 426–464.
- [102] LIU L & SHI DH, 1999, *(s, S) model for inventory with exponential lifetimes and renewal demands*, Naval Research Logistics, **46**, pp. 39–56.
- [103] MACK M, 2012, *Capitalise on freshness*, Supermarket and Retailer, **8**, pp. 13–16.

- [104] MAIHAMI R & NAKHAI KAMALABADI I, 2012, *Joint pricing and inventory control for non-instantaneous deteriorating items with partial backlogging and time and price dependent demand*, International Journal of Production Economics, **136(1)**, pp. 116–122.
- [105] MANAGEMENT STUDY GUIDE, 2016, *Types of retail outlets*, [Online], [Cited October 26th, 2016], Available from %5Curl%7Bhttps://managementstudyguide.com/types-of-retail-outlets.htm%7D.
- [106] MANTRALA MK, LEVY M, KAHN BE, FOX EJ, GAIDAREV P, DANKWORTH B & SHAH D, 2009, *Why is assortment planning so difficult for retailers? A framework and research agenda*, Journal of Retailing, **85(1)**, pp. 71–83.
- [107] MARTÍNEZ-DE-ALBÉNIZ V & ROELS G, 2011, *Competing for shelf space*, Production and Operations Management, **20(1)**, pp. 32–46.
- [108] MARTÍNEZ-ROMERO D, BAILÉN G, SERRANO M, GUILLÉN F, VALVERDE JM, ZAPATA P, CASTILLO S & VALERO D, 2007, *Tools to maintain postharvest fruit and vegetable quality through the inhibition of ethylene action: A review*, Critical Reviews in Food Science and Nutrition, **47**, pp. 543–560.
- [109] MARTIS MS, 2006, *Validation of simulation based models: A theoretical outlook*, Electronic Journal of Business Research Methods, **4(1)**, pp. 39–46.
- [110] MENA C, ADENSO-DIAZ B & YURT O, 2011, *The causes of food waste in the supplier-retailer interface: Evidences from the UK and Spain*, Resources, Conservation and Recycling, **55(6)**, pp. 648–658.
- [111] MENGER K, 1932, *Das Botenproblem*, Ergebnisse eines mathematischen Kolloquiums, **2**, pp. 11–12.
- [112] MILLER CM, SMITH SA, MCINTYRE SH & ACHABAL DD, 2010, *Optimizing and evaluating retail assortments for infrequently purchased products*, Journal of Retailing, **86(2)**, pp. 159–171.
- [113] MITCHELL JE, 2002, *Branch-and-cut algorithms for combinatorial optimization problems*, Oxford University Press, Oxford.
- [114] MLADENOVIĆ N & HANSEN P, 1997, *Variable neighborhood search*, Computers and Operations Research, **24(11)**, pp. 1097–1100.
- [115] MURRAY CC, TALUKDAR D & GOSAVI A, 2010, *Joint optimization of product price, display orientation and shelf-space allocation in retail category management*, Journal of Retailing, **86(2)**, pp. 125–136.
- [116] NAHMAN A & DE LANGE W, 2013, *Costs of food waste along the value chain: Evidence from South Africa*, Waste Management, **33(11)**, pp. 2493–2500.
- [117] NAHMIAS S, 1982, *Perishable inventory theory: A review*, Operations Research, **30(4)**, pp. 680–708.
- [118] NATIONAL RETAIL FEDERATION, 2016, *2016 Top 250 global powers of retailing*, [Online], [Cited February 4th, 2016], Available from %5Curl%7Bhttps://nrf.com/2016/global-250-table%7D.
- [119] NOGALES AF & GOMEZ SUAREZ M, 2005, *Shelf space management of private labels: A case study in Spanish retailing*, Journal of Retailing and Consumer Services, **12**, pp. 205–216.
- [120] OBBAYI SR, 2011, *The major types of database management systems*, [Online], [Cited November 15th, 2016], Available from %5Curl%7Bhttp://www.brighthub.com/interne-t/web-development/articles/110654.aspx%7D.

- [121] OPENBUILDINGS, 2016, *Fairview Green*, [Online], [Cited November 20th, 2016], Available from %5Curl%7Bhttp://openbuildings.com/buildings/fairview-green-profile-38556/media?group=image%7D.
- [122] OPPERMAN R, 2002, *User-interface design*, pp. 233–248 in ADELSBERGER HH & PAWLOWSKI JM (EDS), *Handbook on information technologies for education and training*, Springer, Berlin.
- [123] ORUMIE UC & EBONG D, 2014, *A glorious literature on linear goal programming algorithms*, American Journal of Operations Research, **4**, pp. 59–71.
- [124] PAEZ T, 2009, *Introduction to model validation*, Proceedings of the 27th International Modal Analysis Conference, Orlando (FL), pp. 1–11.
- [125] PARFITT J, BARTHEL M & MACNAUGHTON S, 2010, *Food waste within food supply chains: Quantification and potential for change to 2050*, Philosophical Transactions of the Royal Society B, **365**, pp. 3065–3081.
- [126] PASANDIDEH SHR, NIAKI STA & NIA AR, 2011, *A genetic algorithm for vendor managed inventory control system of multi-product multi-constraint economic order quantity model*, Expert Systems with Applications, **38(3)**, pp. 2708–2716.
- [127] PAUL V, PANDEY R & SRIVASTAVA GC, 2012, *The fading distinctions between classical patterns of ripening in climacteric and non-climacteric fruit and the ubiquity of ethylene: An overview*, Journal of Food Science and Technology, **49(1)**, pp. 1–21.
- [128] PEYTON S, MOSELEY W & BATTERSBY J, 2015, *Implications of supermarket expansion on urban food security in Cape Town, South Africa*, African Geographical Review, **34(1)**, pp. 36–54.
- [129] PIRAMUTHU S & ZHOU W, 2013, *RFID and perishable inventory management with shelf-space and freshness dependent demand*, International Journal of Production Economics, **144(2)**, pp. 635–640.
- [130] PRICEWATERHOUSECOOPERS, 2012, *South African retail and consumer products outlook 2012–2016*, (Unpublished) Technical Report, Cape Town.
- [131] QIN Y, WANG J & WEI C, 2014, *Joint pricing and inventory control for fresh produce and foods with quality and physical quantity deteriorating simultaneously*, International Journal of Production Economics, **152**, pp. 42–48.
- [132] RAAFAT F, 1991, *Survey of literature on continuously deteriorating inventory models*, Journal of the Operational Research Society, **42(1)**, pp. 27–37.
- [133] RACINE JS, 2012, *RStudio: A platform-independent IDE for R and Sweave*, Journal of Applied Econometrics, **27(1)**, pp. 167–172.
- [134] RAMASESHAN B, ACHUTHAN NR & COLLINSON R, 2009, *A retail category management model integrating shelf space and inventory levels*, Asia-Pacific Journal of Operational Research, **26(4)**, pp. 457–478.
- [135] RAMASESHAN B, ACHUTHAN NR & COLLINSON R, 2008, *Decision support tool for retail shelf space optimization*, International Journal of Information Technology and Decision Making, **7(3)**, pp. 547–565.
- [136] RAZ G & PORTEUS EL, 2006, *A fractiles perspective to the joint price/quantity news vendor model*, Management Science, **52(11)**, pp. 1764–1777.
- [137] REYES PM & FRAZIER GV, 2007, *Goal programming model for grocery shelf space allocation*, European Journal of Operational Research, **181**, pp. 634–644.

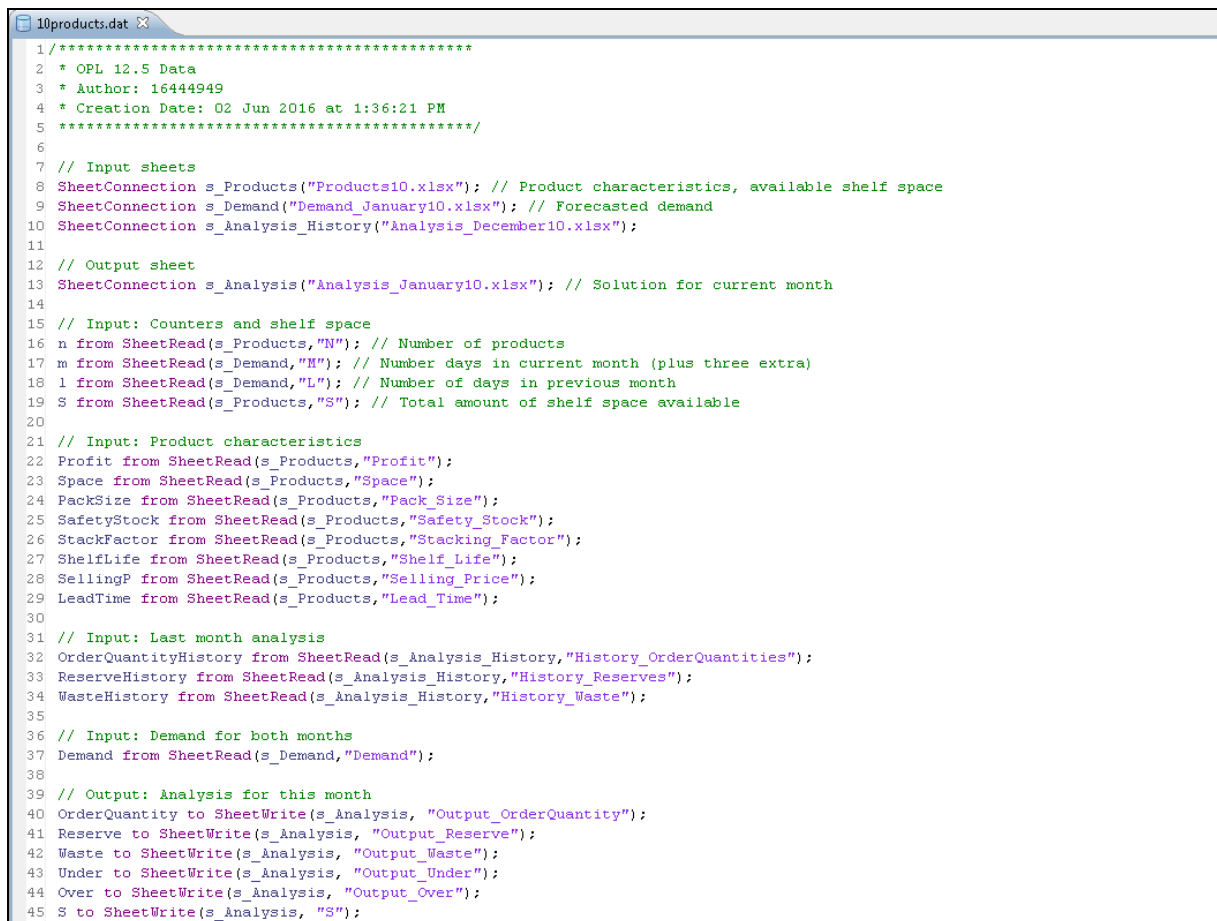
- [138] REYES PM & FRAZIER GV, 2005, *Initial shelf space considerations at new grocery stores: An allocation problem with product switching and substitution*, International Entrepreneurship and Management Journal, **1(254)**, pp. 183–202.
- [139] RIFAI AK, 1996, *A note on the structure of the goal programming model: Assessment and evaluation*, International Journal of Operations and Production Management, **16(1)**, pp. 40–49.
- [140] ROUNDY R, 1985, *98%-effective integer-ratio lot-sizing for one-warehouse multi-retailer systems*, Management Science, **31(11)**, pp. 1416–1430.
- [141] RUSSELL RA & URBAN TL, 2010, *The location and allocation of products and product families on retail shelves*, Annals of Operations Research, **179**, pp. 131–147.
- [142] RYZIN GV & MAHAJAN S, 1999, *On the relationship between inventory costs and variety benefits in retail assortments*, Management Science, **45(11)**, pp. 1496–1509.
- [143] SACHS A, 2014, *Retail analytics: Integrated forecasting and inventory management for perishable products in retailing*, Springer, Cologne.
- [144] SALTVEIT ME, 1999, *Effect of ethylene on quality of fresh fruits and vegetables*, Postharvest Biology and Technology, **15**, pp. 279–292.
- [145] SAURÉ D & ASSAF Z, 2013, *Optimal dynamic assortment planning with demand learning*, Manufacturing and Service Operations Management, **15(3)**, pp. 387–404.
- [146] SCARF HE, 2002, *Inventory theory*, Operations Research, **50(1)**, pp. 186–191.
- [147] SCHLÜNZ EB, 2011, *Decision support for generator maintenance scheduling in the energy sector*, MSc Thesis, Stellenbosch University, Stellenbosch.
- [148] SHIM J, WARKENTIN M, COURTNEY JF, POWER DJ, SHARDA R & CARLSSON C, 2002, *Past, present, and future of decision support technology*, Decision Support Systems, **33(2)**, pp. 111–126.
- [149] SILVER EA, 1981, *Operations research in inventory management: A review and critique*, Operations Research, **29(4)**, pp. 628–645.
- [150] SINGH AK & KAPOOR R, 2013, *A literature review on demand models in retail assortment planning*, International Journal of Marketing and Business Communication, **2(4)**, pp. 1–11.
- [151] SMITH AWJ, POULSTON S, ROWSELL L, TERRY LA & ANDERSON JA, 2009, *A new palladium-based ethylene scavenger to control ethylene-induced ripening of climacteric fruit*, Platinum Metals Review, **53(3)**, pp. 112–122.
- [152] SMITH SA & AGRAWAL N, 2000, *Management of multi-item retail inventory systems with demand substitution*, Operations Research, **48(1)**, pp. 50–64.
- [153] SNYMAN, R, 2015–2016, *Personal Communication*, Contactable at rewillemse@gmail.com.
- [154] TALBI E, 2009, *Metaheuristics: From design to implementation*, John Wiley & Sons, Hoboken (NJ).
- [155] TAVAKKOLI-MOGHADDAM R, RAHIMI-VAHED AR, GHODRATNAMA A & SIADAT A, 2009, *A simulated annealing method for solving a new mathematical model of a multi-criteria cell formation problem with capital constraints*, Advances in Engineering Software, **40(4)**, pp. 268–273.
- [156] TIERSKY E, 2016, *Ethylene and produce: Friends or foes?*, [Online], [Cited November 8th, 2016], Available from <http://shelflifeadvice.com/content/ethylene-and-produce-friends-or-foes%7D>.

- [157] TOPALOGLU H, 2013, *Joint stocking and product offer decisions under the multinomial logit model*, Production and Operations Management, **22(5)**, pp. 1182–1199.
- [158] TRIKI E, COLLETTE Y & SIARRY P, 2005, *A theoretical study on the behavior of simulated annealing leading to a new cooling schedule*, European Journal of Operational Research, **166(1)**, pp. 77–92.
- [159] URBAN TL, 1998, *An inventory-theoretic approach to product assortment and shelf-space allocation*, Journal of Retailing, **74(1)**, pp. 15–35.
- [160] URBAN TL, 2002, *The interdependence of inventory management and retail shelf management*, International Journal of Physical Distribution and Logistics Management, **32(1)**, pp. 41–58.
- [161] VAN DONSELAAR K, VAN WOENSEL T, BROEKMEULEN R & FRANSOO J, 2006, *Inventory control of perishables in supermarkets*, International Journal of Production Economics, **104(2)**, pp. 462–472.
- [162] VAN KAMPEN TJ, VAN DONK DP & VAN DER ZEE D, 2010, *Safety stock or safety lead time: Coping with unreliability in demand and supply*, International Journal of Production Research, **48(24)**, pp. 7463–7481.
- [163] WANG X & LI D, 2012, *A dynamic product quality evaluation based pricing model for perishable food supply chains*, Omega, **40(6)**, pp. 906–917.
- [164] WILSON JG & ANDERSON CK, 2015, *Joint inventory and pricing decisions*, IFAC Proceedings Volumes (IFAC-PapersOnline), **48(3)**, pp. 238–241.
- [165] WINSTON WL & GOLDBERG JB, 2004, *Operations research: Applications and algorithms*, 4th Edition, Brooks/Cole, Belmont (CA).
- [166] YANG M, 2001, *An efficient algorithm to allocate shelf space*, European Journal of Operational Research, **131(1)**, May, pp. 107–118.
- [167] YANG MH & CHEN WC, 1999, *A study on shelf space allocation and management*, International Journal of Production Economics, **60**, pp. 309–317.
- [168] YUCEL E, KARAESMEN F, SALMAN FS & TURKAY M, 2009, *Optimizing product assortment under customer-driven demand substitution*, European Journal of Operational Research, **199**, pp. 759–768.
- [169] ZIUKOV S, 2015, *A literature review on models of inventory management under uncertainty*, Business Systems & Economics, **5(1)**, pp. 26–34.
- [170] ZUFRYDEN FS, 1986, *A dynamic programming approach for product selection and supermarket shelf-space allocation*, Journal of the Operational Research Society, **37(4)**, pp. 413–422.

APPENDIX A

CPLEX source code

This appendix contains the CPLEX source code referred to in Chapter 4. The source code consists of two files. The *.dat* file, displayed in Figure A.1, facilitates the uploading of input data from Excel sheets, and similarly allows for output results to be written to an Excel sheet. In the *.mod* file, displayed in Figures A.2 and A.3, the required variables are initialised and the actual mathematical model is programmed.



```

1 /*****
2 * OPL 12.5 Data
3 * Author: 16444949
4 * Creation Date: 02 Jun 2016 at 1:36:21 PM
5 *****/
6
7 // Input sheets
8 SheetConnection s_Products("Products10.xlsx"); // Product characteristics, available shelf space
9 SheetConnection s_Demand("Demand_January10.xlsx"); // Forecasted demand
10 SheetConnection s_Analysis_History("Analysis_December10.xlsx");
11
12 // Output sheet
13 SheetConnection s_Analysis("Analysis_January10.xlsx"); // Solution for current month
14
15 // Input: Counters and shelf space
16 n from SheetRead(s_Products,"N"); // Number of products
17 m from SheetRead(s_Demand,"M"); // Number days in current month (plus three extra)
18 l from SheetRead(s_Demand,"L"); // Number of days in previous month
19 S from SheetRead(s_Products,"S"); // Total amount of shelf space available
20
21 // Input: Product characteristics
22 Profit from SheetRead(s_Products,"Profit");
23 Space from SheetRead(s_Products,"Space");
24 PackSize from SheetRead(s_Products,"Pack_Size");
25 SafetyStock from SheetRead(s_Products,"Safety_Stock");
26 StackFactor from SheetRead(s_Products,"Stacking_Factor");
27 ShelfLife from SheetRead(s_Products,"Shelf_Life");
28 SellingP from SheetRead(s_Products,"Selling_Price");
29 LeadTime from SheetRead(s_Products,"Lead_Time");
30
31 // Input: Last month analysis
32 OrderQuantityHistory from SheetRead(s_Analysis_History,"History_OrderQuantities");
33 ReserveHistory from SheetRead(s_Analysis_History,"History_Reserves");
34 WasteHistory from SheetRead(s_Analysis_History,"History_Waste");
35
36 // Input: Demand for both months
37 Demand from SheetRead(s_Demand,"Demand");
38
39 // Output: Analysis for this month
40 OrderQuantity to SheetWrite(s_Analysis, "Output_OrderQuantity");
41 Reserve to SheetWrite(s_Analysis, "Output_Reserve");
42 Waste to SheetWrite(s_Analysis, "Output_Waste");
43 Under to SheetWrite(s_Analysis, "Output_Under");
44 Over to SheetWrite(s_Analysis, "Output_Over");
45 S to SheetWrite(s_Analysis, "S");

```

FIGURE A.1: CPLEX source code *.dat* file.

```

10products.mod
1 /*****
2 * OPL 12.5 Model
3 * Author: 16444949
4 * Creation Date: 02 Jun 2016 at 1:36:07 PM
5 *****/
6
7 int n=...; // number of products
8 int m=...; // number of days in current month
9 int l=...; // number of days in previous month
10 int v=m+l; // total number of days in two months
11 int S=...;
12
13 range Products=1..n; // i
14 range Days=1..v; // j
15
16 // Product characteristics
17 float Profit[Products]=...;
18 float Space[Products]=...;
19 float PackSize[Products]=...;
20 float SafetyStock[Products]=...;
21 float StackFactor[Products]=...;
22 int ShelfLife[Products]=...;
23 float SellingP[Products]=...;
24 int LeadTime[Products]=...;
25
26 // Input variables
27 float Demand[Products][Days]=...;
28 int OrderQuantityHistory[Products][1..1]=...;
29 float ReserveHistory[Products][1..1]=...;
30 float WasteHistory[Products][1..1]=...;
31
32 // Decision variables
33
34 dvar float+ OrderQuantity[Products][1..v-3];
35 dvar float+ Reserve[Products][Days];
36 dvar float+ Waste[Products][Days];
37
38 dvar float+ Over[Products][1+1..v];
39 dvar float+ Under[Products][1+1..v];
40
41 // Set CPLEX time limit
42
43 execute{
44   cplex.tilim = 28800
45 }

```

FIGURE A.2: CPLEX source code .mod file — Part 1.

```

46
47 // Model
48
49 // Objective function
50 minimize sum(i in Products, j in Days: j >= 1+1) (Profit[i]*Under[i][j]+SellingP[i]*Waste[i][j]);
51
52 subject to {
53
54 // Historical inputs
55 forall(i in Products, j in Days: 1 <= j <= 1)
56   input_values:
57   {OrderQuantity[i][j] == OrderQuantityHistory[i][j];
58     Reserve[i][j] == ReserveHistory[i][j];
59     Waste[i][j] == WasteHistory[i][j];}
60
61 // Available shelf space
62 forall(j in Days: j >= 1+1)
63   available_shelfspace:
64   sum(i in Products) ((Reserve[i][j]+OrderQuantity[i][j-LeadTime[i]])*PackSize[i])/StackFactor[i] *Space[i] <= S;
65
66 // Demand balance equations
67 forall(i in Products, j in Days: j >= 1+1)
68   demand_balance:
69   Reserve[i][j]+(PackSize[i]*OrderQuantity[i][j-LeadTime[i]])+Under[i][j]-Over[i][j] == Demand[i][j]+ SafetyStock[i];
70
71 // Reserve balance equations
72 forall(i in Products, j in Days: j >= 1+1)
73   reserve:
74   Reserve[i][j] == max1(0,Reserve[i][j-1]+(OrderQuantity[i][j-1-LeadTime[i]])*PackSize[i]-Demand[i][j-1]-Waste[i][j-1]);
75
76 // Waste calculation
77 forall(i in Products, j in Days: j >= 1+1)
78   waste_calculation:
79   Waste[i][j] == max1(0,Reserve[i][j-ShelfLife[i]+1]+OrderQuantity[i][j-ShelfLife[i]-LeadTime[i]+1]*PackSize[i]
80   - sum(a in Days: j-ShelfLife[i]+1 <= a <= j )Demand[i][a] - sum(b in Days: j-ShelfLife[i]+1 <= b <= j-1)Waste[i][b]);
81
82 }

```

FIGURE A.3: CPLEX source code .mod file — Part 2.

APPENDIX B

Simulated annealing algorithm source code

An approximate solution approach by the simulated annealing algorithm for the model of Chapter 3 was presented in Chapter 5. The source code of this algorithm is included in this appendix, and is spread across Figures B.1–B.4

```

# Obtaining a solution in the form of an ordering schedule through simulated annealing
# Janneke Lötter

#####
# 1. Problem specifications
#####

# start the timer
StartTime <- proc.time()

# set problem size and available shelf space
nproducts <- 72
ndays <- 30
maxspace <- 120213

# import products and characteristics
products<-read.csv(file="Products.csv",header = T,row.names = 1)

# import historical data
history_orderquantity<-as.matrix(read.csv(file="History_Orderquantity.csv",header = T,row.names = 1))
history_reserve<-as.matrix(read.csv(file="History_Reserve.csv",header = T,row.names = 1))
history_waste<-as.matrix(read.csv(file="History_Waste.csv",header = T,row.names = 1))
history_under<-as.matrix(read.csv(file="History_Under.csv",header = T,row.names = 1))
history_over<-as.matrix(read.csv(file="History_Over.csv",header = T,row.names = 1))
history_demand<-as.matrix(read.csv(file="History_Demand.csv",header = T,row.names = 1))

# import demand of previous period and current period
demand<-as.matrix(read.csv(file="Demand.csv",header = T,row.names = 1))

#####
# 2. Parameters for simulated annealing
#####

# fixed values
maxiterations <- 300000

maxTemp <- 10165 #p1 a0.2
alpha <- 0.9 # cooling parameter
penalty <- 1 # space violation penalty

acLim <- nproducts*ndays # minimum acceptances before new temperature stage
attemptLim <- 100*nproducts*ndays # number of no-improvement iterations before reheat
eqLim <- 25 # number of times the same cost function value before reheat

# counters
it <- 1 # iteration counter
reheat <- 0 # count number of times to reheat because stuck/attempt
epoch <- 1 # epoch counter
equal <- 0 # count number of times the same cost function value is found
attempt <- 1

maxreheat <- 3

#####
# 3. Create necessary arrays
#####

# order quantity matrix
orders <- matrix(NA,nrow=nproducts,ncol=ndays+1)

# matrix to store previous accepted solution
previousq <- matrix(NA,nrow=nproducts,ncol=ndays+1)
# schedule <- matrix(NA,nrow=nproducts,ncol=ndays)

# matrix to store best solution ordering schedule
BOS <- matrix(NA,nrow=nproducts,ncol=ndays+1)

accept <- array(NA,maxiterations)
temp <- array(NA,maxiterations)

# matrix to store space used per day
used <- matrix(NA,nrow=1,ncol=ndays+2)

# array for cost function values
cost <- array(NA,maxiterations)

# create random numbers for accepting with probability
rand<-runif(maxiterations,0,1)

reheatDetails <- matrix(NA, nrow=maxreheat+1, ncol=3, byrow=FALSE )

output <- data.frame("BestSol","BestSolIt", "It", "Time", "Reheat") #BestSol, BestSolIt, it, RunTime, reheat
write.table(output,file="output.csv",append = TRUE, sep=",",col.names=FALSE,row.names=FALSE)

```

FIGURE B.1: *Simulated annealing algorithm source code — Part 1.*

```

#####
# 4. Initial feasible solution
#####

temp[1] <- maxTemp # T <- MaxTemp

# construct an initial order quantity matrix [PreviousQ <- InitialQ]
# ideal number of packs per product for period (round up)
ideal <- array()
for(i in 1:nproducts){
  ideal[i] <- ceiling( (sum(demand[i,4:(ndays+3)]) + products$Safety.Stock[i]*(ndays/products$Shelf.Life[i])) /
products$Pack.Size[i] )
}

# separate calculation
# fractional ordering schedule (round to nearest int) = first 31 days of initial ordering schedule
# dummy day gets difference between total of round and ideal
for(i in 1:nproducts){
  for(j in 1:ndays){
    previousq[i,j] <- floor(0.5+ (demand[i,j+products$Lead.Time[i]]/products$Pack.Size[i]) ) # round x to nearest
intger in R equals floor (x+0.5)
  }
}

# schedule <- previousq[1:nproducts,1:ndays]
for(i in 1:nproducts){
  previousq[i,ndays+1] <- max(0,ideal[i] - sum(previousq[i,1:ndays]))
}

# create matrices for this month and last month (referred to as double)

orderquantity_double<-cbind(history_orderquantity,previousq) # input data + constructed initial solution

reserve_double<-cbind(history_reserve,matrix(data=NA,nrow=nproducts,ncol=ndays+2,byrow=TRUE))
waste_double<-cbind(history_waste,matrix(data=NA,nrow=nproducts,ncol=ndays+2,byrow=TRUE))
under_double<-cbind(history_under,matrix(data=NA,nrow=nproducts,ncol=ndays+2,byrow=TRUE))
over_double<-cbind(history_over,matrix(data=NA,nrow=nproducts,ncol=ndays+2,byrow=TRUE))
demand_double<-cbind(history_demand,demand)

# fill double matrices with data of this month
for(j in (ncol(history_reserve)+1):(ncol(history_reserve)+ndays+2)){
  for(i in 1:nproducts){
    reserve_double[i,j] <- max(0,reserve_double[i,j-1]+products$Pack.Size[i]*orderquantity_double[i,j-
products$Lead.Time[i]-1]-demand_double[i,j-1]-waste_double[i,j-1])
    waste_double[i,j] <- max(0, reserve_double[i,(j-products$Shelf.Life[i]+1)] +
products$Pack.Size[i]*orderquantity_double[i,(j+1-products$Shelf.Life[i]-products$Lead.Time[i])] - sum(demand_double[i,
(j-products$Shelf.Life[i]+1):j]) - sum(waste_double[i,(j-products$Shelf.Life[i]+1):(j-1)]))
    under_double[i,j] <- max(0,demand_double[i,j]+products$Safety.Stock[i]-
(reserve_double[i,j]+products$Pack.Size[i]*orderquantity_double[i,j-products$Lead.Time[i]]))
    over_double[i,j] <- max(0,reserve_double[i,j]+products$Pack.Size[i]*orderquantity_double[i,j-
products$Lead.Time[i]]-(demand_double[i,j]+products$Safety.Stock[i]))
  }
}

# determine initial cost function value
# loss = unment demand + product waste
loss<-matrix(NA,nrow=nproducts,ncol=ndays+2)
# violation of space constraint
space<-matrix(NA,nrow=nproducts,ncol=ndays+2)
spaceviolation<-matrix(NA,nrow=1,ncol=ndays+2)

# unsatisfied demand and product waste
for(i in 1:nproducts){
  for(j in 1:(ndays+2)){
    loss[i,j]<-
products$Profit[i]*under_double[i,j+ncol(history_under)]+products$Selling.Price[i]*waste_double[i,j+ncol(history_waste)]
  }
}

# shelf space used by each product each day, violation where more than max space
for(j in 1:(ndays+2)){
  for(i in 1:nproducts){
    space[i,j]<- products$Space[i]*
(reserve_double[i,j+ncol(history_under)]+products$Pack.Size[i]*orderquantity_double[i,j+ncol(history_under)-
products$Lead.Time[i]])/products$Stacking.Factor[i]
  }

  used[j] <- sum(space[,j])
  spaceviolation[j] <- max(0,used[j]-maxspace) # determine violation of each day
}

# total cost (losses + penalty for space violation)
cost[1] <- sum(loss) + penalty*sum(spaceviolation)

```

FIGURE B.2: *Simulated annealing algorithm source code — Part 2.*

```

BestSol <- cost[1]
BestSolIt <- 1

#####
#                               Simulated annealing
#####

while (reheat<=maxreheat & it < maxiterations){
  it <- it+1

  # if min acceptances made, cool system, new epoch
  if ((sum(accept!="rejected",na.rm = TRUE)) >= (acLim * epoch)){
    temp[it] <- alpha*temp[it-1]
    epoch <- epoch + 1
    attempt <- 0
  }

  # if min acceptances not yet made, keep temperature
  else{
    temp[it] <- temp[it-1]
    attempt <- attempt + 1
  }

  # if system frozen => new temp (reheat)
  if (attempt == attemptLim | equal == eqLim){
    reheat <- reheat + 1

    if (attempt == attemptLim){
      reheatDetails[reheat,] <- c(reheat, it, "Max attempts" )
    }

    if (equal == eqLim){
      reheatDetails[reheat,] <- c(reheat, it, "Equal" )
    }

    equal <- 0
    attempt <- 0
    if(reheat == 1){temp[it] <- maxTemp*0.4} #0.4
    if(reheat == 2){temp[it] <- maxTemp*0.25}
    if(reheat == 3){temp[it] <- maxTemp*0.1}
  }

  # determine cost of this iteration's solution
  # generate new neighbour      [ApplyMoveOperator(CurrentQ)]

  orders <- previousq # copy previous accepted solution  [CurrentQ <- PreviousQ]

  z<-ceiling(runif(1,0,nproducts))
  a<-ceiling(runif(1,0,ndays+1))
  b<-ceiling(runif(1,0,ndays+1))

  if (orders[z,a]>0 & orders[z,b]>0 | orders[z,a]==0 & orders[z,b] > 0){
    orders[z,a] <- orders[z,a] + 1
    orders[z,b] <- orders[z,b] - 1
  }
  else if (orders[z,a]>0 & orders[z,b]==0){
    orders[z,a] <- orders[z,a] - 1
    orders[z,b] <- orders[z,b] + 1
  }
  else{
  }

  orderquantity_double<-cbind(history_orderquantity,orders)

  # CaluculateCost (CurrentQ)
  # determine all the matrices
  for(j in (ncol(history_reserve)+1):(ncol(history_reserve)+ndays+2)){
    for(i in 1:nproducts){
      reserve_double[i,j] <- max(0,reserve_double[i,j-1]+products$Pack.Size[i]*orderquantity_double[i,j-
products$Lead.Time[i]-1]-demand_double[i,j-1]-waste_double[i,j-1])
      waste_double[i,j] <- max(0, reserve_double[i, (j-products$Shelf.Life[i]+1)] +
products$Pack.Size[i]*orderquantity_double[i, (j+1-products$Shelf.Life[i]-products$Lead.Time[i])] - sum(demand_double[i,
(j-products$Shelf.Life[i]+1):j]) - sum(waste_double[i, (j-products$Shelf.Life[i]+1):(j-1)])) )
      under_double[i,j] <- max(0,demand_double[i,j]+products$Safety.Stock[i]-
(reserve_double[i,j]+products$Pack.Size[i]*orderquantity_double[i,j-products$Lead.Time[i]]))
      over_double[i,j] <- max(0,reserve_double[i,j]+products$Pack.Size[i]*orderquantity_double[i,j-
products$Lead.Time[i]]-(demand_double[i,j]+products$Safety.Stock[i]))
    }
  }

  # unsatisfied demand and product waste
  for(n in 1:nproducts){
    for(d in 1:(ndays+2)){
      loss[n,d]<-
products$Profit[n]*under_double[n,d+ncol(history_under)]+products$Selling.Price[n]*waste_double[n,d+ncol(history_waste)]

```

FIGURE B.3: *Simulated annealing algorithm source code — Part 3.*

```

    }
  }

  # shelf space used by each product each day, violation where more than max space
  for(d in 1:(ndays+2)){
    for(n in 1:nproducts){
      space[n,d]<- products$Space[n]*
      (reserve_double[n,d+ncol(history_under)]+products$Pack.Size[n]*orderquantity_double[n,d+ncol(history_under)-
      products$Lead.Time[n]])/products$Stacking.Factor[n]
    }

    used[d] <- sum(space[,d]) # sum over each column to obtain space used per day
    spaceviolation[d] <- max(0,used[d]-maxspace) # determine violation of each day

  }
  # total cost (losses + penalty for space violation)
  cost[it] <- round(sum(loss) + penalty*sum(spaceviolation),2)

  # if solution is same as previous, increase 'equal' by one; else, start counting from 0 again
  if (cost[it] == cost[it-1]){
    equal <- equal +1
  }
  else{
    equal <- 0
  }

  # accept improving solution
  if (cost[it] < cost[it-1]){
    accept[it] <- "accepted:improvement" # keep track of iteration
    previousq <- orders
    if (cost[it] < BestSol){
      BestSol <- cost[it] # store improving solution as best solution
      BestSolIt <- it # store number of this iteration
      BOS <- orders # store best ordering schedule
    }
  }

  # accept non-improving solution with probability
  else if (rand[it-1]<= exp((cost[it-1]-cost[it])/temp[it])){
    accept[it] <- "accepted:probability" # keep track of iteration
    previousq <- orders
  }
  # otherwise reject non-improving solution
  else {
    accept[it] <- "rejected" # keep track of iteration
    cost[it] <- cost[it-1] # store previous iteration's cost function value
  }

  cat("\r",it, it/maxiterations*100)
  flush.console()
}
RunTime <- proc.time() - StartTime

output <- data.frame(BestSol, BestSolIt, it, RunTime[3], reheat)
write.table(output,file="output.csv",append = TRUE, sep=",",col.names=FALSE,row.names=FALSE)

```

FIGURE B.4: *Simulated annealing algorithm source code — Part 4.*

APPENDIX C

Input data for the hypothetical problem

This appendix contains data referred to in both Chapters 4 and 5. A hypothetical problem instance involving 10 products was described in §4.2. Forecasted demand data for these 10 products for both the current decision period and the previous decision period were randomly generated, and may be found in Tables C.1 and C.2, respectively. A historical replenishment order schedule was generated so as to satisfy the forecasted demand for the previous decision period approximately, and is shown in Table C.3. The other required historical input data were calculated in `Microsoft Excel`, based on the historical replenishment order schedule. The historical inventory values are shown in Table C.4, followed by the historical underachievement- and overachievement values in Tables C.5 and C.6. The historical waste values are finally presented in Table C.7.

TABLE C.1: Product demand for an extended decision period of 34 days for all ten products of the hypothetical problem instance.

Day	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
Product																	
1	36	46	22	6	29	49	40	37	35	41	41	52	44	67	51	100	108
2	19	6	8	10	3	14	17	3	0	2	4	13	6	7	8	17	8
3	14	37	9	56	30	59	4	100	50	31	30	31	69	75	46	25	35
4	3.65	2.64	2.37	3.02	2.25	3.01	3.21	4.65	4.44	3.42	2.83	2.39	1.57	1.89	5.26	3.39	3.91
5	5	6	3	2	8	4	9	7	11	6	9	5	2	2	2	3	11
6	2	5	3	9	10	5	4	2	2	2	4	5	1	5	8	7	9
7	1	2	2	2	2	3	3	2	3	5	1	2	5	4	4	1	5
8	0.60	0.21	0.19	0.58	0.30	2.01	1.36	1.17	0.91	0.33	0.54	0.30	0.76	0.58	1.13	0.32	0.46
9	2	3	3	3	3	4	2	1	2	1	1	5	2	4	1	1	6
10	0.35	2.27	0.96	2.87	0.18	1.96	1.08	1.47	3.66	1.80	0.37	3.00	1.43	0.85	1.75	2.77	1.05
Day	18	19	20	21	22	23	24	25	26	27	28	29	30	31	1	2	3
Product																	
1	100	77	72	94	53	84	69	13	11	7	34	60	55	69	60	55	69
2	24	14	8	3	12	10	0	4	6	14	8	3	12	10	3	12	10
3	44	30	30	57	85	40	82	64	39	51	91	87	31	52	87	31	52
4	3.25	1.6	0.62	2.39	2.28	1.82	4.08	1.56	2.82	3.22	1.9	0.45	0.71	0.21	0.45	0.71	0.21
5	4	5	10	4	1	2	7	1	4	3	6	4	8	5	4	8	5
6	5	6	4	4	5	7	8	10	5	2	3	5	5	8	5	5	8
7	3	6	2	2	2	1	1	3	3	5	3	3	1	4	3	1	4
8	0.16	1.84	0.72	0.89	0.76	1.19	1.52	0.55	0.84	0.16	0.2	0.65	0.68	1.82	0.65	0.68	1.82
9	3	3	2	2	2	2	1	4	1	2	2	1	1	2	1	1	2
10	0.40	1.46	1.04	1.12	0.77	0.38	2.85	0.60	0.62	1.28	1.55	1.90	1.27	3.59	1.90	1.27	3.59

TABLE C.2: Historical product demand for the decision period for all ten products of the hypothetical problem instance.

Day	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Product																
1	21	36	29	36	46	22	6	29	49	40	37	35	41	41	52	44
2	4	6	8	19	6	8	10	3	14	17	3	0	2	4	13	6
3	68	48	35	14	37	9	56	30	59	4	100	50	31	30	31	69
4	1.06	0.74	0.71	3.65	2.64	2.37	3.02	2.25	3.01	3.21	4.65	4.44	3.42	2.83	2.39	1.57
5	2	10	4	5	6	3	2	8	4	9	7	11	6	9	5	2
6	4	5	6	2	5	3	9	10	5	4	2	2	2	4	5	1
7	2	10	6	1	2	2	2	2	3	3	2	3	5	1	2	5
8	0.45	0.66	0.69	0.6	0.21	0.19	0.58	0.3	2.01	1.36	1.17	0.91	0.33	0.54	0.3	0.76
9	1	4	8	2	3	3	3	3	4	2	1	2	1	1	5	2
10	1.16	0.96	2.48	0.35	2.27	0.96	2.87	0.18	1.96	1.08	1.47	3.66	1.8	0.37	3	1.43
Day	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	
Product																
1	67	51	100	108	100	77	72	94	53	84	69	13	11	7	34	
2	7	8	17	8	24	14	8	3	12	10	0	4	6	14	8	
3	75	46	25	35	44	30	30	57	85	40	82	64	39	51	91	
4	1.89	5.26	3.39	3.91	3.25	1.6	0.62	2.39	2.28	1.82	4.08	1.56	2.82	3.22	1.9	
5	2	2	3	11	4	5	10	4	1	2	7	1	4	3	6	
6	5	8	7	9	5	6	4	4	5	7	8	10	5	2	3	
7	4	4	1	5	3	6	2	2	2	1	1	3	3	5	3	
8	0.58	1.13	0.32	0.46	0.16	1.84	0.72	0.89	0.76	1.19	1.52	0.55	0.84	0.16	0.2	
9	4	1	1	6	3	3	2	2	2	2	1	4	1	2	2	
10	0.85	1.75	2.77	1.05	0.4	1.46	1.04	1.12	0.77	0.38	2.85	0.6	0.62	1.28	1.55	

TABLE C.4: Historical inventory values for the decision period for all ten products of the hypothetical problem instance.

Day	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Product																
1	0	0	0	0	24	38	16	10	0	11	31	54	79	98	117	125
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	21	25	25	26	52	12	22	21	21	20
4	0	0	0	0	0	9.36	6.99	3.97	1.72	0	8.79	4.14	0	8.58	5.75	3.36
5	0	0	0	0	9	3	0	0	6	2	1.47	8.47	6.55	0	5	0
6	0	0	0	0	0	0	0	9	17	12	8	24	22	20	34	29
7	0	0	0	0	0	6	4	10	8	5	2	0	0	3	2	0
8	0	0	0	0	0	0	0	0	7.7	5.69	1.57	0.41	0	0	0	0
9	0	0	0	0	0	0	0	0	0	4	2	1	7	6	5	8
10	0	0	0	0	0	0	0	0	0	6.63	5.54	4.08	0.42	0	0	0
Day	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	
Product																
1	141	134	143	163	115	135	118	106	72	79	55	46	33	22	15	
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	86	
3	11	0	14	19	14	0	0	30	0	0	20	0	0	0	0	
4	13.8	11.9	6.65	3.26	0	0	0	0	9.61	7.33	5.51	13.4	11.9	9.05	5.83	
5	0	0	0	0	1.55	0	0	4	0	0	0	7	6	10.5	7.46	
6	28	23	15	8	0	0	0	0	14	9	2	12	20	15	13	
7	3	7	11	10	13	10	12	10	8	14	13	12	9	6	9	
8	0	0	0	0	0	0	6.16	5.44	4.55	3.79	2.6	1.08	0	0	0	
9	6	10	9	8	10	15	12	18	16	14	12	11	15	14	12	
10	0	0	0	0	0	0	0	0	10.9	10.1	9.73	6.88	6.28	5.66	4.38	

TABLE C.6: Historical overachievement values for the decision period for all ten products of the hypothetical problem instance.

Day	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Product																
1	0	0	0	0	0	0	0	0	0	0	9	34	53	72	80	96
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	22	0	0	0	0	0	0
4	0	0	0	0	5.36	2.99	0	0	0	4.79	0.14	0	4.58	1.75	0	9.8
5	0	0	0	3	0	0	0	0	0	1	2.47	5.47	0	0	0	0
6	0	0	0	0	0	0	7	15	10	6	22	20	18	32	27	26
7	0	0	0	0	3	1	7	5	2	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	5.7	3.69	2.33	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	1	0	0	4	3	2	5	3
10	0	0	0	0	0	0	0	0	8.04	3.54	2.08	0	0	0	0	0
Day	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	
Product																
1	89	98	118	70	90	73	61	27	34	10	1	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0	0	0	77	69	0
3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	7.91	2.65	0	0	0	0	0	5.61	3.33	1.51	9.43	7.87	5.05	1.83	0	0
5	0	0	0	0	0	0	0	0	0	0	1	0	10	1.46	0	0
6	21	13	6	0	0	0	0	12	7	0	10	18	13	11	8	0
7	4	8	7	10	7	9	7	5	11	10	9	6	3	6	3	0
8	0	0	0	0	0	4.16	3.44	2.55	1.79	0.6	0	0	0	0	0	0
9	7	6	5	7	12	9	15	13	11	9	8	12	11	9	7	0
10	0	0	0	0	0	0	0	8.88	8.11	7.73	4.88	4.28	3.66	2.38	0.83	0

APPENDIX D

Additional data for the case study

Additional data required for the case study of Chapter 7 are included in this appendix. The demand for the 72 products sold in the ambient section of the fresh produce department of the retail outlet in Grassy Park over a one-month decision period is provided in Table D.1, while the historical demand for these products pertaining to the previous decision period may be found in Table D.2. Furthermore, Table D.3 contains the historical replenishment order schedule and Table D.4 contains the historical inventory values. Finally, the historical waste values are provided in Table D.5.

The two benchmark solutions referred to in §7.4.1, namely the initial solution constructed and the replenishment order schedule that was actually employed at the retail outlet over the same time period, may be found in Tables D.6 and D.7, respectively.

TABLE D.3: Historical replenishment order schedule for the 72 products for the decision period prior to the period considered in the case study.

Product	Day																														
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
1	1	0	3	2	3	4	5	3	2	3	2	2	3	3	1	0	3	4	1	5	0	0	1	4	0	2	6	0	4	1	2
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1
3	0	1	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0	1	0	0	2	1	0	1	1
5	1	1	0	0	0	0	1	0	0	1	0	1	0	1	0	0	1	1	0	0	0	0	1	0	0	1	1	1	1	1	0
6	0	0	0	1	0	1	0	0	0	1	0	2	1	1	0	2	1	0	2	0	1	0	1	2	2	0	4	0	1	2	1
7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	1	0	0	1	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1
9	0	0	1	0	0	0	1	0	1	0	0	1	0	0	0	1	0	0	0	0	1	0	1	0	0	1	0	0	0	1	0
10	0	0	0	0	0	1	1	1	1	1	1	1	1	1	0	1	1	0	1	0	0	1	0	1	0	0	0	0	0	1	0
11	0	0	0	1	1	0	0	1	0	1	0	0	0	0	1	0	2	0	1	0	0	0	0	1	0	0	0	0	0	1	0
12	1	2	0	0	0	0	0	1	2	1	2	0	0	2	0	0	1	2	3	1	1	4	2	0	3	0	1	1	2	0	0
13	1	1	0	1	1	0	0	1	0	2	1	0	0	0	0	2	0	0	1	1	0	0	3	1	1	1	2	1	1	3	1
14	0	0	0	0	1	0	0	0	1	0	0	0	0	1	0	0	1	0	0	1	0	0	0	1	0	1	0	1	1	0	0
15	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
16	1	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	1	1
17	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	3	0	1
18	1	0	0	0	0	1	0	0	1	0	0	0	0	0	1	1	0	1	1	0	0	0	3	0	0	0	0	2	2	1	1
19	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	1	1	1
20	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
21	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	1	0	0	0	0	1	0	0	1	0
22	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
23	2	2	1	2	3	3	2	2	2	2	1	2	1	3	3	3	2	3	2	0	3	4	3	3	6	1	1	4	0	2	1
24	0	1	0	1	0	0	0	1	0	1	1	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	1	0	0	0
25	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0
26	0	1	0	1	0	0	1	0	2	1	1	1	0	1	0	0	0	1	0	1	2	0	0	2	1	2	2	1	3	0	0
27	1	0	0	0	5	3	0	3	1	1	2	4	3	1	4	1	8	0	0	8	1	0	0	1	4	0	2	3	4	1	1
28	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
29	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
30	0	0	0	0	1	1	1	1	2	3	0	1	0	2	0	1	0	2	1	1	1	0	3	0	0	0	0	2	1	0	1
31	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0
32	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2
33	1	0	0	0	0	0	0	1	0	0	0	1	0	2	0	0	0	1	0	0	1	2	1	0	4	1	1	2	0	1	1
34	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
35	0	1	0	5	2	1	2	3	0	1	0	1	0	1	0	2	0	0	1	1	0	1	0	2	1	2	0	1	0	1	1
36	1	0	2	0	1	0	1	2	0	1	0	4	1	0	0	1	0	1	3	0	0	1	1	0	1	1	1	4	1	0	0
37	0	3	0	0	2	0	3	1	0	2	1	3	1	1	2	1	2	0	1	3	2	1	1	2	6	3	7	1	0	1	2
38	3	3	2	1	0	3	1	1	1	2	1	1	0	2	1	5	3	1	1	1	2	1	0	1	2	2	0	1	2	2	1
39	0	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
40	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0
41	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	1	1	0	1	0	1
42	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
43	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
44	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
45	1	0	6	6	0	2	2	1	2	1	0	1	1	2	0	6	2	2	0	6	1	0	0	3	0	0	4	3	0	1	2
46	1	0	0	2	6	2	1	1	4	0	7	1	1	4	1	0	3	6	1	2	6	1	4	5	3	1	5	0	4	4	4
47	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
48	0	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	2

TABLE D.3: (continued) Historical replenishment order schedule for the 72 products for the decision period prior to the period considered in the case study.

Product	Day																															
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	
49	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	
50	2	7	3	7	4	1	4	5	0	4	8	3	8	7	9	1	17	3	0	16	0	3	10	2	9	1	11	0	9	0	2	
51	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	
52	0	0	0	0	2	0	0	0	1	0	1	0	0	1	0	0	0	0	1	2	0	0	0	0	2	0	0	1	0	1	0	
53	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	
54	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	
55	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	1	1	
56	1	1	1	0	0	1	0	0	0	2	0	1	0	2	1	0	0	0	0	1	1	0	0	0	0	1	3	2	0	1		
57	0	0	1	0	0	0	0	0	1	0	1	1	0	0	0	1	0	0	1	0	0	0	0	0	1	0	0	2	1	1		
58	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	
59	0	0	0	0	1	0	1	1	1	0	0	1	0	0	1	0	1	0	1	0	1	2	3	0	0	2	1	0	5	1	1	
60	0	0	0	3	1	0	0	1	1	1	0	1	3	0	1	1	0	3	0	2	2	1	1	0	0	1	0	1	3	1	1	
61	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
62	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
63	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
64	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	2	0	0	0	1	0	1	0	0	1	0	1	0	0	0
65	1	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	1	
66	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	0	1	
67	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	1	0	0	1	0	1	0	0	0	1	0	0	0	1	1	
68	0	1	1	0	0	0	0	0	1	0	0	1	0	0	1	0	0	0	1	1	1	1	1	0	0	0	0	0	1	0	1	
69	1	0	0	0	1	0	0	2	1	0	1	0	1	1	0	0	1	1	1	0	0	1	0	0	4	0	0	2	1	1		
70	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	1	0	0	1	0	0	0	0	1	0	1	0	1	0	1	1	
71	0	0	0	0	1	0	1	0	1	2	0	1	0	0	0	1	0	0	0	1	0	0	1	0	2	0	0	0	0	1	1	
72	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	

TABLE D.4: Historical inventory values for the 72 products during the decision period prior to the period considered in the case study.

Product	Day																															
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	
1	0	0	0	0	0	0	1.3	1.6	2.9	5.2	7.59	7.98	7.38	7.77	7.17	6.56	6.95	7.88	6.8	4.72	5.65	7.57	6.49	9.42	6.24	3.06	0.88	1.7	0	0	2.82	
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	7.24	6.48	5.71	4.95	4.19	3.32	2.46	1.59	0	0	0	0	
3	0	0	0	0	0	7.69	7.38	7.07	6.76	14.45	13.96	21.46	16	15.5	15.01	14.51	8	7.54	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	10.9	21.9	32.4	27.8	23.3	18.8	14.3	9.73	5.21	0.45	0	10.2	5.49	0.73	25.9	36.2		
5	0	0	0	0	9.2	18.4	12.59	6.79	0.99	0	0	9.63	4.25	0	0	9.63	4.25	14.84	10.44	6.03	16.62	27.21	22.8	18.39	13.99	9.59	5.19	15.78	11.38	6.98	17.6	
6	22.7	55.2	87.8	74.7	61.7	48.7	35.6	37.6	24.6	26.5	13.9	1.2	0	2.34	0	17.3	19.7	19.9	5.19	20.4	20.7	5.94	21.2	6.44	8.77	0	2.33	19.7	37	24.3	71.7	
7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	5.09	4.18	3.28	8.37	13.5	12	11.1	10.2	6	0	0	0	0	0	5.07	4.14	3.22	2.29	1.36	6	5.29	4.57	3.86	3.15	0	0	0	0
9	0	0	0.81	0.62	0.42	0.22	1.03	0.83	0.63	0.44	1.1	0.76	1.42	1.08	0.73	1.39	1.05	0.57	0.09	0.61	0.13	0	0	0	0.61	0.23	0.84	0.46	0.07	0.69	0.3	
10	46.5	50.2	53.9	54.5	41.2	27.8	14.5	1.09	0	0.64	0.4	0.16	0	0	0	0	0	0.72	0	0.72	1.44	0	0.72	0	0	3.18	0	3.18	0	0	0	0
11	0	0	0	0	0	0	0	4.19	8.37	6.56	5.34	10.13	8.91	12	10.78	9.56	6	4.18	6	4.18	14.36	12.54	16.72	14.89	13.34	6	4.44	6	4.44	2.88	1.32	
12	239	251	263	273	283	300	240	220	200	180	158	167	205	210	240	180	159	181	144	106	98.9	122	174	167	171	266	300	275	339	314	319	
13	0	0	0	0	8.93	17.85	8.78	17.71	26.63	17.56	6.71	13.85	3	28.15	35.29	24.44	13.59	3.6	0	26.02	16.03	6.05	14.06	22.08	10.8	0	42.72	49.45	56.17	62.89	87.61	
14	0	0	0	0	0	0	0	10.6	7.2	4.65	2.11	13.6	11	8.49	5.95	0	11.6	9.13	6.69	18.3	14	25.6	23.1	14	8.92	0	8.92	3.83	12.8	7.66		
15	0	0	4.64	3.23	1.81	0.39	0	0	0	0	0	11.45	10.9	10.35	9.8	9.26	8.71	8.18	7.65	7.12	0	0	0	0	0	0	0	0	0	0	0	0
16	0	0	0	0	6.12	4.25	10.4	8.49	6.62	4.74	3.03	1.32	0	6.29	4.58	2.87	1.16	0	0	0	0	0	0	0	5.98	12	9.93	7.91	5.89	3.87	1.84	
17	59.1	82.5	85.9	110	114	109	103	97.7	91.2	86.3	84.5	80	50	40	10	0	0	0	0	0	0	0	0	0	0	7.76	5.51	3.27	1.03	0	0	0
18	8.75	11.1	11.5	10.9	11.3	10.6	10	9.41	8	8	6	3	2.35	1.7	1.05	0.4	0.35	1.3	2.24	2.19	3.14	4	3.95	3.34	2.74	5.13	4.53	3.93	3.32	2.72		
19	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	10.02	8.03	6.03	16.03	14.02	12.02	10.02	8.02	6.01	4.05	2.09	0.12	0	0	10
20	12.5	12.2	11.9	11.6	11.3	10	9.71	9.43	19.1	10	9.45	8.9	8.35	7.8	7.25	16.7	16.1	15.5	14.8	14.1	13.4	12.7	12	11.4	11.3	11.2	11.1	11	11	10.8	10.7	
21	0	0	0	0	0	1.67	0.35	0	0	0	0	0	0	0	0	3	3	2.86	2.71	2.57	2.43	2.29	5.14	5	4.04	6	5.04	4.08	3.12	2.16	4.2	
22	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
23	2.52	2.78	3.04	4.43	3.81	3.2	1.58	0.96	1.35	1.73	1.36	0.99	0.61	0.24	0	0	0	0.67	1.34	2.02	1.69	2.36	2.03	0	0.6	2.2	2.8	3.41	7.01	5.61	4.21	
24	0	0	0	0	0	6.52	5.04	11.6	10.1	8.6	7.15	13.7	12.3	16	22.6	21.1	19.7	16	14.6	8	0	0	0	0	5.98	3.95	1.93	7.9	5.88	3.85	1.83	

TABLE D.4: (continued) Historical inventory values for the 72 products during the decision period prior to the period considered in the case study.

Product	Day	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
25	11.13	10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	8.81	7.62	6.43	5.24	4.05	2.86	1.67	0.76	19.85	18.94	18.03	17.12	16.21	15.3
26	0	0	0	0	0	9.12	4.23	13.4	8.46	3.58	9.88	2.18	22.5	28.8	35.1	41.4	33.7	33.6	19.5	5.36	0	0	0	0	20.2	12.4	4.6	24.8	31	51.2	71.4	
27	74.5	81.29	73.09	88.84	74.59	45.34	16.09	0	45.75	61.5	29.33	42.16	25	7.83	5.66	33.5	46.33	20.89	40.46	15.02	94.58	54.15	13.71	93.27	77.1	45.93	14.76	0	28.83	0	0	
28	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	14.5	13.9	13.4	12.9	12.5	12.1	11.7	11.3	10.8	10.5	0
29	0	0	13.31	26.77	26.23	25.7	25.16	24.63	24.09	23.56	37.19	36.82	28	14	13.63	13.27	12.9	12.58	12.27	11.95	0	0	0	0	0	0	0	0	0	0	0	0
30	22.3	41.9	43.5	45.8	30.1	14.4	0	0	2.28	4.55	2.86	1.16	17.5	51.8	32.1	30.4	10.7	29.3	11.9	12.6	0	18.6	19.3	19.9	19.9	1.92	37.9	19.9	1.94	0	0	
31	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5.88	5.76	5.63	5.51	5.39	5.27	5.15	
32	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
33	437	519	541	631	660	600	570	540	420	360	240	240	120	60	22.6	45.3	7.93	97.2	66.4	35.7	4.93	34.2	3.43	0	28.6	117	146	114	323	352	380	
34	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
35	4.17	7.46	6.75	6.86	5.96	6.06	5.17	9.27	10.38	10.48	11.77	14.06	13.35	13.63	12.92	13.21	12.5	12.76	11	11	10	8	6	6.26	5.05	4.84	3.62	4.41	4.2	4.99	3.78	
36	20	10	30	22.7	25.4	18.1	30.7	23.4	26.1	18.8	20.4	32.1	23.7	25.4	17	48.7	50.3	42.1	33.9	35.7	27.4	29.2	51	43	36.1	39.5	42.9	36.3	39.7	43.1	46.5	
37	1.72	1.02	1.32	3.13	1.94	3.75	2.56	1.37	2.19	1	2.34	1.68	0.03	0.37	0	1.34	0.69	0.38	1.07	0.77	1.46	0.15	0	1.69	0.89	0	0	0	3.19	3.38	7.58	
38	9.25	33.5	27.8	45.9	64.1	82.2	85.4	73.6	46.7	64.9	52.8	40.7	28.6	31.6	19.5	7.37	0	0	0	0	43.5	57	40.6	24.1	7.61	11.9	1.32	0	0	4.35	8.71	0
39	0	0	0	0	0	0	0	0	0	0	23.4	21.8	20.2	18.59	16.99	40.39	38.79	37.52	36.24	34.97	33.69	32.42	31.14	29.87	28.58	27.28	25.99	24.7	23.4	22.11	20.8	
40	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	7.48	6.95	6.32	5.69	5.05	4.42	3.79	3.15	2.52
41	0	0	0	0	0	0	0	0	0	14.44	11.75	9.05	6.36	3.67	0.97	0	0	14.48	10.95	7.43	3.9	18.38	14.86	12.3	9.73	7.17	4.61	0	15.44	30.9		
42	0	0	0	14.4	13.8	13.1	12.5	11.9	11.3	10.7	10.1	24.5	23.9	23.4	15	14.4	13.9	13	12.2	11.4	10.6	9.82	15	14.2	13.3	12.4	11.4	10.5	9.61	8.69	7.78	
43	0	0	0	0	23.56	22.11	20.67	19.22	17.78	16.33	14.94	13.55	12.16	10.78	9.39	8	6.61	5.3	3.98	2.67	1.36	25.05	23.74	22.42	21.13	19.84	18.56	17.27	15.98	14.69	13.4	
44	0	5.68	5.35	5.09	4.82	4.56	10.3	10	9.76	9.5	9.06	8.63	8.2	6	5.57	5.14	10.7	10.3	6	5.56	5.11	4.67	4.22	3.78	3.49	3.21	2.93	2.64	0	0	0	
45	0	0	0	0	0	0	3.87	7.73	5.6	5.47	5.34	4.2	4.07	2.94	0.81	0	0	0	0	3.87	3.73	3.6	1.47	5.34	6.2	6.07	5.93	8.79	8.66	4	7	
46	41.7	38.6	52.5	64.7	25.9	0	0	0	46.3	24.5	0	0	15.1	0	66.1	30.3	0	16.7	0	0	50.7	16.4	0	48.3	11.5	25.8	57.1	54.3	17.7	48.9		
47	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5.77	5.55	5.32	5.09	4.87	4.64	4.41	4.22	4.03	3.83	3.64	3.45	3.26	3.07	
48	0	0	0	0	0	0	7.01	6.03	5.04	4.15	3.26	0	0	0	0	0	0	6.9	5.81	4.71	3.62	2.52	0	6.9	5.37	3.84	2.31	0.78	0	0	0	
49	12.97	12.63	12.28	11.94	11.59	11.25	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
50	0.97	2.17	5.37	6.25	4.13	7.01	5.89	8.77	8.65	5.53	5.69	6.85	3.01	3.18	7.34	6.5	10.7	10.7	12.8	6.92	17	13.1	6.17	15.3	11.2	10.2	16.2	14.1	19.1	16	23	
51	59.46	78.49	73.52	71.26	60	57.74	60	48	36	33.74	32.24	30.74	12	10.5	9	7.5	6	12	11.05	10.11	9.16	8.22	7.27	6.33	4.36	2.38	0.41	10.44	8.47	6.5	4.52	
52	0	0	0	5.94	3.87	1.81	0	0	13.9	11.9	9.52	7.16	8	5.64	11.3	8.93	6.58	11.7	8	5.07	2.15	0	5.07	18.2	16	13.9	11.7	0	13.9	11.7	9.54	
53	0	0	0	0	0	0	0	0	0	0	0	0	0	7.61	7.22	6.83	6.44	6.06	0	0	0	0	0	0	0	0	0	0	0	0	0	0
54	12.9	11.9	10.8	9.49	8.18	6.86	5.54	4.23	0	0	14.7	13.5	12.2	10.9	9.66	8.4	7.13	5.74	4.35	2.96	0	0	0	0	0	0	0	0	0	15.7	15.4	15.1
55	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5.03	4.07	3.1	2.13	1.3	0.47	0	0	0	0	0	0
56	3.21	0.66	6.1	10.2	14.3	18.4	14.5	10.5	14.6	10.7	5.29	0	10.6	5.13	7.7	2.27	12.8	16.3	11.7	7.11	2.54	0	0	3.43	7.4	3.38	0	0	0	3.98	24	
57	0	0	0	0	0	0	5.37	2.73	0.1	0	0	0	5.19	2.39	7.58	12.78	9.97	7.01	4.05	9.09	6.13	11.17	8.21	5.26	2.11	0	0	0	0	4.85	1.71	
58	0	0	0	0	0	0	0	7.2	6.4	5.3	4.21	3.12	2.03	0.93	0	0	0	0	0	0	0	0	6.64	5.28	4.23	3.19	2.14	1.09	0.05	0	0	
59	3.91	15.83	27.74	23.41	19.07	14.73	10.39	14.05	9.71	13.38	17.09	20.81	16.53	12.25	15.96	11.68	7.4	9.65	3.9	6.14	0.39	2.64	0	2.25	9.74	25.22	16.71	8.2	15.69	15.18	6.66	
60	0	0	6.03	5.05	0	0	15	14	5.06	0	0	0	0	0	0	14.5	3.67	0.84	0	0	13.2	2.34	7.51	14.4	13.3	12.2	3.1	0	0	0	0	
61	0	0	0	0	0	0	0	0	0	0	0	0	0	5.97	5.95	5.92	5.89	5.87	0	0	0	0	0	0	0	0	0	0	0	0	0	0
62	0	0	0	0	0	0	7.18	6.37	5.55	4.73	4.15	3.57	2.98	2.4	0	0	0	0	0	0	7.29	6.58	5.87	5.17	4.46	3.8	3.15	2.5	0	0	0	0
63	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	10.48	4.96	0	0	10.48	4.96	0	0	0	0	0	0	0	0	0	0
64	16	21.6	27.2	25.7	24.2	22.7	21.1	19.6	18.1	16.6	15.9	8	0	0	0	7.25	6.49	2.91	0	4.42	16.8	13.3	9.66	6.08	11.6	9.05	14.5	12	9.51	15	12.5	
65	0	0	0	0	10.14	8.28	6.42	4.56	14.71	12.85	11.04	9.24	7.43	5.63	15.82	14.02	12.21	10.33	8.45	6.57	4.69	2.81	12.93	11.04	8.9	6.75	4.6	2.45	12.3	10.15	8	
66	0.02	0	0	0	0	0	0	0	0	0	0	0	0	6.9	5.79	12.7	11.6	11.6	11.6	11.6	11.6	11.6	11.6	11.6	11.6	8	6.69	5.39	4.08	2.78	1.47	
67	0	11.99	15.98	11.95	7.91	3.87	0	0	0	0	0	4.42	0.84	0	0	4.42	0.84	0	0	3.23	0	0	3.23	0	4.88	1.75	0	0	4.88	1.75	0	
68	0	0	0	0	0	5.42	10.9	8.27	5.69	3.12	0.28	0	5.16	2.32	0	5.16	2.32	0.01	5.71	3.41	1.1	0	5.7	11.4	17.5	23.5	21.6	19.7	16	8	0	
69	27.3	22.61	33.91	36.88	39.85	32	26.97	16	16	8	2.82	13.64	16.47	11.29	14.11	8.93	11.75	14.17	8.6	3.02	5.44	7.86	10.28	12.71	8.41	4.12	7.83	3.54	0	27.71	23.4	
70	0	5.41	10.8	8.16	5.49	2.82	0.16	0	0	0	0	0	0	5.8	11.6	9.4	7.34	5.28	11.2	8	5.94	11.9	9.83	7.56	5.29	3.02	8	5.73	11.5	9.19		
71	3.9	17.81	31.71	26.64	21.58	16.52	11.45	0	4.94	0	5.1	0.21	5.31	20.41	15.52	20.62	15.72															

TABLE D.5: *Historical waste values for the 72 products during the decision period prior to the period considered in the case study.*

Product	Day	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.72	0	0	0	0	
3	0	0	0	0	0	0	0	0	0	0	0	0	4.97	0	0	0	6.02	0	7.09	0	0	0	0	0	0	0	0	0	0	0	0	0	
4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
8	0	0	0	0	0	0	0	0	0	0.55	0	0	3.29	5.1	0	0	0	0	0	0	0	0	0	0.43	0	0	0	0	2.44	0	0	0	
9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
11	0	0	0	0	0	0	0	0	0	0	0	0	1.69	0	0	2.35	0	2.36	0	0	0	0	0	0	0	5.78	0	2.88	0	0	0	0	
12	0	0	0	0	22.7	39.9	0	0	0	0	0	0	3.92	8.5	38.5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
13	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
14	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3.4	0	0	0	0	0	1.82	0	0	4.04	0	3.83	0	0	0	0	0	
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	6.59	0	0	0	0	0	0	0	0	0	0	0	
16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
17	0	0	0	0	0	0	0	0	0	0	0	2.62	28.15	8.15	28.15	8.15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
18	0	0	0	0	0	0	0	0.79	0.38	1.35	2.35	0.35	0	0	0	0	0	0	0	0	0	0	0.09	0	0	0	0	0	0	0	0	0	
19	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
20	0	0	0	0	1.01	0	0	0	0	8.85	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
21	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.08	0	0	0	0	0	0	
22	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
23	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
24	0	0	0	0	0	0	0	0	0	0	0	0	2.81	0	0	0	2.22	0	5.13	6.57	0	0	0	0	0	0	0	0	0	0	0	0	
25	0.1	8.97	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
26	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
27	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
28	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
29	0	0	0	0	0	0	0	0	0	0	0	0	8.46	13.63	0	0	0	0	0	0	11.64	0	0	0	0	0	0	0	0	0	0	0	
30	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
31	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
32	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
33	0	0	0	1.89	30.49	0	0.97	90.49	30.49	82.64	22.64	82.64	22.64	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
34	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
35	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1.02	1.26	0.26	1.26	2.26	0	0	0	0	0	0	0	0	0	
36	2.4	2.4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
37	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
38	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
39	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
40	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1.92
41	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
42	0	0	0	0	0	0	0	0	0	0	0	0	0	0	7.79	0	0	0	0	0	0	0	9.01	0	0	0	0	0	0	0	0	0	0
43	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
44	0	0	0	0	0	0	0	0	0	0	0	0	1.77	0	0	0	0	0	3.82	0	0	0	0	0	0	0	0	0	2.36	0	0	0	0
45	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4.52	0.86	0	0
46	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
47	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
48	0	0	0	0	0	0	0	0	0	0	0	0	2.38	0	0	0	0	0	0	0	0	0	1.42	0	0	0	0	0	0	0	0	0	0

TABLE D.5: (continued) Historical waste values for the 72 products during the decision period prior to the period considered in the case study.

Product	Day																														
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
49	0	0	0	0	0	10.9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
50	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
51	0	0	0	9	0	7.48	9.74	9.74	0	0	0	17.24	0	0	0	0	5.05	0	0	0	0	0	0	0	0	0	0	0	0	0	0
52	0	0	0	0	0	0	0	0	0	0	4.8	0	0	0	0	0	0	0.72	0	0	0	0	0	0	0	0	9.54	0	0	0	
53	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5.67	0	0	0	0	0	0	0	0	0	0	0	0	0	
54	0	0	0	0	0	0	2.91	0	0	0	0	0	0	0	0	0	0	0	1.57	0	0	0	0	0	0	0	0	0	0	0	
55	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
56	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
57	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
58	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
59	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
60	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
61	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5.84	0	0	0	0	0	0	0	0	0	0	0	0	0	
62	0	0	0	0	0	0	0	0	0	0	0	0	1.82	0	0	0	0	0	0	0	0	0	0	0	0	1.85	0	0	0		
63	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
64	0	0	0	0	0	0	0	0	0	7.1	7.25	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
65	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
66	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3.59	0	5.39	0	0	0	0	0	0	
67	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
68	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1.76	6.07	6.07	0	
69	0	0	0	0	2.82	0	5.94	2.97	2.97	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
70	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1.17	0	0	0	0	0	0.75	0	0	0	0	
71	0	0	0	0	0	0	6.39	0	0	0	0	0	0	0	0	0	0	0	0	1.17	0	0	0	0	0	0	0	0	0	0	
72	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	9.89	0	0	0	0	0	0	

TABLE D.6: The initial solution constructed for the case study.

Product	Day																														
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
1	1	1	1	1	1	3	2	1	1	1	1	3	1	0	0	1	0	1	2	1	1	1	1	1	1	1	2	1	1	1	4
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	1	0	1	1	0	0	0	0	0	0	5
3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4
4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	9
5	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	1	0	0	0	0	0	1	1	0	0	0	0	0	0	6
6	1	1	1	1	1	1	1	1	2	2	0	1	0	1	1	1	1	2	1	1	1	1	2	3	1	1	1	1	1	0	2
7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5
9	0	0	0	0	0	0	0	0	1	0	0	0	1	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	6
10	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0
11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	8
12	1	1	1	1	1	1	1	1	2	1	1	1	1	1	2	2	1	1	1	1	1	1	2	2	1	1	1	1	1	1	5
13	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	9
14	0	0	0	1	0	0	1	0	0	1	1	1	0	0	0	1	1	0	0	0	0	0	0	1	1	0	0	0	0	0	8
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	8
16	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	1	1	0	0	0	0	8
17	1	1	1	0	1	0	1	1	0	0	0	0	0	1	1	1	0	0	0	1	1	1	1	1	0	0	0	1	1	1	1
18	1	1	1	0	1	1	0	0	1	1	0	1	1	0	0	1	1	0	1	1	0	0	0	1	0	1	0	0	0	0	2
19	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4
20	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2
21	0	0	0	0	1	0	0	0	1	0	0	1	1	1	1	1	1	1	0	1	0	1	0	1	0	1	0	0	0	0	6
22	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
23	2	2	2	2	2	3	2	2	2	3	1	3	3	2	1	1	3	1	3	3	3	1	2	4	1	3	2	2	1	1	19
24	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	7

TABLE D.6: (continued) The initial solution constructed for the case study.

Product	Day																															
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	
25	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4
26	1	1	1	1	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	5	
27	2	2	2	2	3	2	2	2	3	3	2	2	2	2	2	3	3	2	2	2	2	1	3	3	2	2	1	2	1	1	9	
28	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	
29	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	
30	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	3	
31	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	
32	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	
33	1	1	1	0	1	1	1	0	1	0	1	1	1	1	1	1	1	1	1	0	1	0	1	0	1	1	1	1	1	1	0	
34	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	
35	1	1	1	1	1	1	0	0	1	1	1	0	1	0	0	0	1	0	1	0	1	0	0	0	0	1	0	1	0	0	3	
36	1	1	1	1	1	0	0	1	1	1	1	0	0	0	1	1	1	1	0	0	0	1	0	1	1	1	0	0	1	1	4	
37	2	2	2	2	2	3	3	1	5	4	3	2	3	4	2	5	6	3	2	3	4	2	6	6	4	3	4	6	3	3	5	
38	1	1	1	1	2	2	2	1	2	3	1	2	2	2	1	2	3	1	2	1	2	1	2	2	1	1	1	1	1	1	9	
39	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	
40	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	
41	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	10	
42	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	
43	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	
44	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	
45	1	1	1	1	1	1	2	1	1	1	1	1	1	2	1	1	1	0	0	0	1	1	1	1	1	0	0	1	0	0	8	
46	4	4	4	4	4	4	5	3	4	6	3	3	3	4	2	3	5	3	4	3	4	3	3	3	5	3	3	2	3	2	31	
47	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	
48	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	6	
49	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	
50	4	4	4	3	3	2	4	5	4	3	2	3	2	7	8	7	6	3	3	2	4	5	4	3	2	3	2	4	4	4	29	
51	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	9	
52	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	12	
53	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	
54	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	6	
55	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	7	
56	0	0	0	0	1	1	1	1	1	2	3	1	1	1	1	1	2	3	1	1	1	1	1	1	2	3	1	1	1	1	6	
57	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	8	
58	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	
59	1	1	1	1	2	2	2	1	1	1	2	3	2	2	1	1	1	3	3	3	2	1	1	1	3	1	3	2	1	1	1	
60	1	1	1	1	1	1	1	1	2	2	1	1	1	1	1	2	1	1	1	1	1	1	2	1	1	1	1	1	1	1	5	
61	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	
62	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5	
63	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	1	1	0	0	0	0	0	6	
64	0	0	0	0	0	0	1	0	0	0	0	1	0	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1	1	5	
65	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	6	
66	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	6	
67	0	0	0	0	1	1	0	0	0	0	1	1	0	0	0	0	0	0	1	1	0	0	0	0	1	1	1	1	0	0	4	
68	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	1	1	0	0	0	0	0	1	1	0	0	0	0	0	0	4	
69	1	1	1	0	0	1	1	0	1	1	0	0	0	0	0	1	1	0	0	0	0	0	1	1	0	0	0	0	0	0	7	
70	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	1	1	0	0	0	0	0	1	1	0	0	0	0	0	0	5	
71	0	0	0	0	0	0	1	1	1	1	0	0	0	1	0	1	0	0	0	0	0	0	1	0	0	0	0	1	0	0	9	
72	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5	

TABLE D.7: *The replenishment order schedule actually employed in the retail outlet in Grassy Park.*

Product	Day																														
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	
1	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	1	1	0	1	1	2	
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
4	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0	1	1	0	1	
5	0	0	0	0	0	1	1	0	0	1	0	0	0	1	1	0	0	1	0	0	1	1	0	1	0	0	0	1	0	0	
6	1	0	1	0	1	1	2	2	1	0	0	1	1	1	2	0	0	0	0	0	2	2	1	1	0	0	0	1	1	1	
7	0	0	0	0	0	1	0	1	0	0	0	0	1	0	1	0	0	1	0	0	1	1	0	0	0	0	0	0	0	0	
8	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	1	
10	1	0	0	0	0	0	0	0	0	1	1	1	1	1	1	0	0	0	0	0	1	0	0	0	1	1	1	1	1	1	
11	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	
12	2	1	1	1	1	1	2	1	1	1	1	1	2	2	2	1	2	0	1	1	2	2	1	1	0	1	2	1	1	1	
13	1	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	1	0	1	0	0	0	
14	1	1	0	0	0	0	0	0	1	0	1	1	0	0	1	1	1	0	0	0	0	0	1	1	0	0	1	0	0	1	
15	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	1	1	
16	0	1	0	0	0	0	1	0	1	1	1	0	0	0	1	0	1	1	0	0	0	0	0	1	0	0	1	0	0	0	
17	0	1	1	1	1	1	0	0	1	1	0	1	1	1	1	0	0	1	1	0	1	0	1	0	0	0	0	1	1	1	
18	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	1	0	3	
19	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	
20	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
21	1	0	0	0	0	0	0	0	1	1	0	1	1	1	0	0	0	0	1	0	1	0	1	1	1	1	1	0	0	2	
22	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
23	2	0	0	0	0	0	0	3	1	2	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	6	
24	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	
25	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
26	0	1	1	0	0	0	0	1	0	1	0	1	1	1	0	1	0	0	2	0	0	1	0	0	0	0	0	0	0	1	
27	2	2	2	1	2	1	3	3	2	3	1	2	1	3	4	2	3	2	2	1	3	3	2	2	1	2	2	2	0	0	
28	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
29	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
30	1	0	0	0	0	1	2	1	1	0	1	1	1	1	1	1	1	1	1	0	0	1	1	0	0	1	1	1	1	1	
31	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
32	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
33	0	1	1	0	0	1	0	0	0	0	0	1	1	1	1	1	1	1	0	0	0	1	1	0	0	1	1	1	1	1	
34	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	
35	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	1	
36	0	0	10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
37	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	3	2	4	4	2	2	2	3	3	9	9	6	
38	2	2	2	1	1	2	3	3	1	2	1	2	1	2	3	2	2	1	2	1	2	3	1	1	1	1	1	1	0	0	
39	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
40	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	
41	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	
42	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	1	0	0
43	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	
44	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
45	1	1	1	1	0	1	1	1	1	1	1	1	1	0	1	1	0	0	0	1	0	1	1	0	0	1	1	1	0	2	
46	5	4	4	4	4	3	4	6	4	4	4	5	3	3	4	3	4	3	4	3	4	3	5	3	3	2	4	2	2	4	6
47	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	
48	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	

TABLE D.7: (continued) The replenishment order schedule actually employed in the retail outlet in Grassy Park.

Product	Day																													
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
49	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
50	0	0	0	0	0	5	0	0	0	0	0	0	0	0	0	0	0	0	4	5	0	0	0	0	2	5	4	4	3	4
51	0	1	1	0	0	1	1	0	0	1	0	1	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0	1	0
52	1	1	0	0	0	0	1	1	1	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	1	0	1	1	0	
53	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
54	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	1
55	0	0	0	1	0	1	0	0	0	1	1	1	1	0	1	0	0	1	0	0	1	0	1	0	0	0	0	1	1	0
56	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	3	0	0	0	1	1	2	0
57	0	0	0	1	1	1	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	1	1	0
58	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1
59	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	1	2	1	1	0	1	1	2
60	2	1	0	1	0	0	1	2	1	1	1	0	1	1	0	0	0	0	0	1	2	2	1	1	1	1	1	1	1	0
61	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0
62	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
63	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1	1
64	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	1	2	1	1	3
65	0	0	1	1	0	1	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0
66	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
67	0	0	1	0	0	0	0	0	0	0	1	1	0	1	0	0	1	1	1	0	0	0	1	1	1	1	0	0	0	1
68	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	1	0	0	0	1	1	0	0	0	0	1	1	0
69	1	0	1	1	1	0	1	1	0	1	1	0	1	1	1	0	1	0	1	0	0	1	0	0	1	0	0	0	1	1
70	0	0	1	0	0	0	1	1	0	0	0	0	1	0	1	0	0	0	0	0	0	1	0	0	0	0	0	1	0	1
71	1	1	1	0	1	0	0	1	0	1	1	0	1	0	0	1	0	0	0	0	0	0	1	1	0	1	0	0	0	1
72	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0