Layout and Operations Management of Distribution Centers for Perishables

R.A.C.M. Broekmeulen
Stellingen

behorende bij het proefschrift

Layout and Operations Management of Distribution Centers for Perishables

van

Rob A.C.M. Broekmeulen
I

De opslag van een assortiment verse groenten en fruit bestaande uit minimaal de tien meest verkochte soorten in Nederland vereist minimaal drie temperatuurzones in de vers-distributiecentra.

Dit proefschrift

II

Het uitblijven van een tijd-temperatuur indicator op consumentenverpakkingen van bederfelijke producten komt omdat het grootwinkelbedrijf de condities in de aanvoerketen niet kan garanderen.


III

De verbeterde bewaarfaciliteiten voor hardfruit resulteren in speculatie bij de telers in plaats van een langere houdbaarheid van het product voor de consument.

VERMEULEN, M. [1998], Speculerende telers laten zelf appels en peren bederven, *Volkskrant* 16 april, pag. 2.

IV

De reactie van klanten tegen het einde van de markt op het afprijzen van bederfelijke producten maakt de individuele verschillen in gevoeligheid voor prijs en resterende houdbaarheid bij deze ‘ramphandel’ goed zichtbaar.
V

Het hoge percentage uitval van verse producten bij de consument wordt veroorzaakt door onvoldoende kennis over de gewenste bewaarcondities en door overschatting van de relatie tussen uiterlijke kenmerken en de houdbaarheid van het product.


VI

De bedrijven met kantoren zonder vaste werkplekken voor de werknemers beweren dat niet het verhogen van de bezettingsgraad van de werkplekken maar het bevorderen van de informatieuitwisseling de belangrijkste reden is voor deze experimentele kantoren. Indien deze communicatie-experimenten met afwisselende contacten op het werk slagen dan pleit dit ook voor frequentere kantinebezoek in traditionele kantoren.


PENNEKAMP,E. [1997], Meer, beter en sneller met minder, Arbeidsomstandigheden Concreet 6, Februari, 28–30.

VII

Langdurig werklozen vergroten hun kans op een baan als de potentiële werkgever weet waarom ze zolang geen werk konden krijgen.


VIII

Bestudering van de niet-coderende gedeelten van het humaan DNA laat zien dat de evolutie net zo onestructureerd programmeert als de mensen die deze evolutie heeft voortgebracht.
IX

Slimme logistieke concepten verlagen pas de milieubelasting van het transport als ze domme gewoonten in voldoende mate kunnen verdringen.


X

Men mag zich pas verplaatsen met een eigen auto als men aantoonbaar niet beschikt over het benodigde organisatietalent voor het gebruik van het openbaar vervoer.
Layout and Operations Management of Distribution Centers for Perishables
CIP-DATA LIBRARY TECHNISCHE UNIVERSITEIT EINDHOVEN

Broekmeulen, Robertus Alphonsus Cornelis Maria

Layout and operations management of distribution centers for perishables / by Robertus Alphonsus Cornelis Maria Broekmeulen.
- Proefschrift. - ISBN 90-386-0689-3
NUGI 694
Keywords: Warehousing / Perishables / Distribution

printed by Ponsen & Looijen, Wageningen
cover photo by Jan Vogels

©1998 by R.A.C.M. Broekmeulen, Nuenen, the Netherlands.

All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, photocopying, recording or otherwise, without prior permission of the author.

The research described in this thesis was performed at the Agrotechnological Research Institute ATO-DLO, Wageningen, the Netherlands, and was partly financed by the Dutch Commodity Board for Vegetables and Fruits.
Layout and Operations Management of Distribution Centers for Perishables

PROEFSCHRIFT

ter verkrijging van de graad van doctor aan de Technische Universiteit Eindhoven, op gezag van de Rector Magnificus, prof.dr. M. Rem, voor een commissie aangewezen door het College voor Promoties in het openbaar te verdedigen op maandag 29 juni 1998 om 16.00 uur

door

Robertus Alphonsus Cornelis Maria Broekmeulen

egenoten te Beek (L.)
Dit proefschrift is goedgekeurd
door de promotoren

prof.dr. E.H.L. Aarts
en
prof.dr. A.G. de Kok

en de copromotor
dr.ir. G.A.L. van de Vorst
Preface

For someone who started his career in molecular genetics, becoming an expert in the distribution of perishables cannot be achieved without the help of many persons. I want to express my gratitude to some of them.

First of all, I want to thank Peter Reinders for convincing me that I should pursue a Ph.D. degree. Being an assistant professor at the time, I did not have enough time to write a thesis. After offering me a research position at the Agrotechnological Research Institute ATO-DLO, he also gave me for many years the opportunity to work on this topic.

As a geneticist, I was taken with the idea of applying genetic algorithms to solve the problems concerning the distribution of fresh produce. I want to thank Emile Aarts for his willingness to supervise a research topic that used local search mainly as a tool. During the research his early scepticism about the use of genetic algorithms proved to be true. As a relative outsider to the field of distribution and perishables, he never hesitated to question practices and procedures that seemed illogical at first sight. His encouragements to investigate these practices have improved my work considerably. Finally, Emile has taught me much on how to report on research in a proper way.

I also appreciated the interest of Ton de Kok in my work. He immediately understood the need for quality change models when investigating the distribution of perishables.

Besides Emile and Ton, I am indebted to Alfred van de Vorst and Huub ten Eikelder for carefully reading drafts of my thesis. I thank Clare Wilkinson for giving me support in the use of her native language. The \LaTeX\ gurus Wim Nuijten, Mark Sloof, Marco Verhoeven, and Kees van Weert are acknowledged for supplying me with all the necessary software and instructions to produce this thesis.

I am grateful to the management of the companies that supplied me with data and practical insight in the distribution of perishables. I hope that
the results of this thesis will help them with their operations in the future. Special thanks go to my former colleagues at ATO-DLO, in particular Anneke Polderdijk, Henry Boerrigter and Pol Tijskens. They taught me all the different aspects of the quality of fresh produce, and how to model keeping quality. I enjoyed working at the Department of Logistics and Marketing but I can understand why many of my former co-workers have left the department during the last year. Furthermore, I thank my family and friends for their continuing support and interest in my work. Mom, I am sure that you will be delighted about my academic achievements, and Dad, I hope you are convinced that this research has practical use. Finally, I thank my girlfriend Désirée for her support and care especially when the completion of this thesis seemed to take forever. She makes me realize each day that you do not need mathematical models of perishables to keep food fresh in a refrigerator.

Nuenen, April 1998

Rob Broekmeulen
Contents

1 Introduction
   1.1 Physical distribution ........................................ 1
   1.2 Problem formulation ........................................... 2
   1.3 Towards solving the problem .................................... 4
      1.3.1 Decision support system .................................. 6
      1.3.2 Hierarchical planning .................................... 7
      1.3.3 Implementation of ADEPT ................................. 10
   1.4 Thesis outline ................................................ 11

2 Distribution of perishables ........................................ 13
   2.1 Definition and properties ..................................... 13
   2.2 Production and distribution ................................... 17
   2.3 Integral quality control ....................................... 19
   2.4 Perishables in a distribution center ......................... 20
      2.4.1 Activities .................................................. 21
      2.4.2 Resources .................................................. 22
      2.4.3 Operations management ................................. 25
   2.5 Case examples ................................................ 29

3 Slot planning ...................................................... 35
   3.1 The warehouse slot planning problem ....................... 35
      3.1.1 Handling models ........................................ 37
      3.1.2 Definition of the WSPP ............................... 47
<table>
<thead>
<tr>
<th>3.1.3</th>
<th>Problem analysis</th>
<th>48</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1.4</td>
<td>The forward-reserve problem</td>
<td>49</td>
</tr>
<tr>
<td>3.1.5</td>
<td>The stock location assignment problem</td>
<td>53</td>
</tr>
<tr>
<td>3.1.6</td>
<td>Complexity analysis</td>
<td>55</td>
</tr>
<tr>
<td>3.1.7</td>
<td>Decomposition strategies</td>
<td>57</td>
</tr>
<tr>
<td>3.2</td>
<td>The WSPP for perishables</td>
<td>60</td>
</tr>
<tr>
<td>3.2.1</td>
<td>Quality change models</td>
<td>61</td>
</tr>
<tr>
<td>3.2.2</td>
<td>Definition of the WSPP for perishables</td>
<td>69</td>
</tr>
<tr>
<td>3.2.3</td>
<td>Problem analysis</td>
<td>71</td>
</tr>
<tr>
<td>3.2.4</td>
<td>The assignment problem for perishables</td>
<td>71</td>
</tr>
<tr>
<td>3.2.5</td>
<td>Decomposition strategy</td>
<td>74</td>
</tr>
<tr>
<td>3.2.6</td>
<td>Problem extensions</td>
<td>75</td>
</tr>
<tr>
<td>3.3</td>
<td>Overview of the hierarchical decomposition strategy</td>
<td>77</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>4</th>
<th>Local search</th>
<th>79</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.1</td>
<td>Introduction</td>
<td>79</td>
</tr>
<tr>
<td>4.2</td>
<td>Implementation</td>
<td>83</td>
</tr>
<tr>
<td>4.2.1</td>
<td>Repeated descent</td>
<td>83</td>
</tr>
<tr>
<td>4.2.2</td>
<td>Tabu search</td>
<td>84</td>
</tr>
<tr>
<td>4.2.3</td>
<td>Simulated annealing</td>
<td>84</td>
</tr>
<tr>
<td>4.2.4</td>
<td>Parallel descent</td>
<td>84</td>
</tr>
<tr>
<td>4.2.5</td>
<td>Genetic algorithms</td>
<td>85</td>
</tr>
<tr>
<td>4.3</td>
<td>Constraints</td>
<td>86</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>5</th>
<th>Assignment of perishables to zones</th>
<th>87</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.1</td>
<td>Solution strategy</td>
<td>87</td>
</tr>
<tr>
<td>5.2</td>
<td>Problem generation</td>
<td>89</td>
</tr>
<tr>
<td>5.3</td>
<td>Initial solutions and lower bounds</td>
<td>93</td>
</tr>
<tr>
<td>5.4</td>
<td>Computational results</td>
<td>99</td>
</tr>
</tbody>
</table>
Contents

5.4.1 Algorithmic performance 100
5.4.2 Discussion 101
5.5 Problem extensions 115
  5.5.1 Variable zone temperature 116
  5.5.2 Layout redesign 118
5.6 Operational assignment 121
  5.6.1 Simulation model of storage and retrieval 121
  5.6.2 Results of the simulation study 123
5.7 Conclusions 125

6 Forward-Reserve Allocation 127
  6.1 Introduction 127
  6.2 Problem generation 131
  6.3 Solution strategy 136
    6.3.1 Construction of initial solutions 136
    6.3.2 Neighborhood function 137
    6.3.3 Upper bound 138
    6.3.4 Reduction of the problem size 138
  6.4 Computational results 139

7 Stock Location Assignment 145
  7.1 Solution strategies 145
    7.1.1 Handling oriented strategy 146
    7.1.2 Space oriented strategy 149
  7.2 Computational results 153
    7.2.1 Problem generation 153
    7.2.2 Algorithmic performance 154
    7.2.3 Discussion 155
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>8 Conclusions and recommendations</td>
<td>159</td>
</tr>
<tr>
<td>Bibliography</td>
<td>162</td>
</tr>
<tr>
<td>List of symbols</td>
<td>170</td>
</tr>
<tr>
<td>Index</td>
<td>175</td>
</tr>
<tr>
<td>Samenvatting</td>
<td>177</td>
</tr>
<tr>
<td>Curriculum Vitae</td>
<td>180</td>
</tr>
</tbody>
</table>
Introduction

In this thesis we are concerned with the layout and operations management of distribution centers for perishables. Perishables from agricultural origin such as fresh foodstuffs form a major and still increasing part of the consumer goods. Perishables need special attention during physical distribution because of their perishable nature.

Distribution centers for perishables have received little attention in the inventory and distribution management literature despite their economic importance. The focus of this thesis is the layout and operations management of a distribution center for perishables of agricultural origin. The characteristics of perishables make the layout and operations management of a distribution center for perishables more complex than that of a typical distribution center. The turnover rate of products that are handled in such a distribution center is usually very high. The average retention time of a product is slightly more than one day. The types of packaging differ a lot because of a lack in standardization in the distribution chain. The supply and demand of perishables are seasonal, and they fluctuate strongly from day to day. Most perishables need special storage conditions and careful handling to minimize keeping quality loss. Product interactions such as hormone or odor transmission have to be avoided for the same reason.
This research tries to integrate the specific properties of perishables in the models and algorithms that deal with layout and operations management problems of distribution centers. We have developed a decision support system that incorporates the developed models and algorithms. With the decision support system, the management can handle the specific problems of a distribution center for perishables.

1.1 Physical distribution of perishables

Physical distribution includes all activities associated with the shipment of products from the manufacturer to the consumer. The chain of activities between the production and the market is called the distribution chain. Physical distribution is a fundamental aspect of the logistic management of distribution chains. Well-organized physical distribution is seen nowadays by authors such as Porter [1985] and Coyle, Bardi & Langley [1996] as a way of staying competitive in markets that are dominated by buyers. To support physical distribution a wide range of methods, techniques and tools have been developed. An introduction is given by Tompkins & Harmerlink [1994]. The efficiency of a distribution center plays a crucial role in the distribution chain.

A distribution center is a building with two main functions: warehousing and distribution. Warehousing includes all activities concerned with storage and retrieval of products. The storage accommodation is the part of the distribution center where products can be stored. The storage equipment such as rack and shelving systems creates in the storage accommodation locations and storage space where products can be stored. We denote the combination of a fixed location with an allocated storage space as a slot. A warehouse is a building specific for the warehousing function.

The distribution function in a distribution center concentrates on the activities groupage and shipment of customer orders. A customer order is a request of a specific customer for a specific quantity of a specific product at a specific time with an agreed delivery time. Groupage combines one or more customer orders in a single consignment. With handling we denote all labour related activities such as putaway, replenishing, and order picking. Handling equipment such as carts, forklift trucks, conveyors, and sorters handles the products. A storage policy determines where a product can be stored. A retrieval policy determines from which slot the
retrieval should take place. Storage and retrieval policies are examples of operating policies. According to Van den Berg [1996], the combination of storage equipment, handling equipment and operating policies is called a warehousing system. A distribution center may use a set of different warehousing systems. The layout defines the size and the location of the different warehousing systems in the storage accommodation.

Activities associated with value added logistics such as repackaging and grading of the products are frequently carried out in a distribution center but are considered by us as additional services compared to warehousing and distribution, and are therefore not included in our research.


Perishables of agricultural origin are products that are susceptible to keeping quality loss such as aging and breakdown. Vegetables, fruits, meat and dairy products are examples. The distribution of perishables usually starts after the harvest and includes activities such as transportation, handling in and out of storage and conditioning. An article or stock keeping unit (SKU) is one of the appearances of a product. Variation in cultivar, quality class, country of origin, and packaging define different articles of the same product. The assortment of a distribution center is a list of the articles that are handled or in stock during a specific period of time. A year can be divided in one or more planning periods. The supply of many perishables depends on the seasons. The mismatch between the long, seasonal production pattern and the short, day-to-day fluctuating consumption pattern makes the stock levels in a distribution center unpredictable. The numerous changes in the assortment during the year make the activities in a distribution center non-repetitive.
In this thesis we assume that a distribution center for perishables mainly stores and handles perishables. An important warehousing activity in a distribution center for perishables is the conditioning of products. A storage room or cold store is a separate part of the storage accommodation with specific storage conditions such as relative humidity and temperature. An air-conditioned storage accommodation and several storage rooms with insulation distinguish the layout of a distribution center for perishables from a standard distribution center according to Sims [1994]. The specific layout and operations management of a distribution center for perishables influence the amount of keeping quality loss and the efficiency of the operations. According to Hoogerwerf, Reinders, Oosterloo & Kanis [1990], the share of distribution cost in the consumer price of perishables is nearly twice as high as the share of distribution cost in the consumer price of non-perishables. This difference is due to two complicating factors, i.e., fast handling and special storage conditions. Chung & Norback [1991] notice that perishables are distributed in a relatively short time to minimize the keeping quality loss. Adequate storage conditions, minimal handling and avoidance of product interactions such as odor and hormone transmission can reduce keeping quality loss as is illustrated in Ryall & Lipton [1979] and Petropakis [1989]. Meffert [1990] describes the trade-off between conditioning and fast handling, and remarks that it is difficult to improve the current speed of handling because it is already high. The importance of a distribution center in a distribution chain is explained further in Chapter 2.

1.2 Problem formulation

The management of a distribution center for perishables has to deal with questions concerning layout and operations management. The following list mentions some of the important issues without pretending to be complete.

- What kind of storage and handling equipment is needed?
- What kind of storage policies should be applied?
- How much storage space is required for the different warehousing systems?
- Where should the warehousing systems be located in the layout of the distribution center?
- What kind of storage conditions are needed to reduce the keeping
1.2. Problem formulation

quality loss of the articles?
- Where must the articles be placed in the distribution center and does the location depend on the time of year?
- How much personnel is required?

The problem associated with answering these questions can be modeled in a quantitative way in terms of the decision quantities, the objectives, and the constraints. An instance of the problem can be defined by a dataset that describes the building, the available equipment, the expected customer orders, and the properties of the articles in the assortment. The customer orders determine the flow of articles through the distribution center. The minimal required information about the expected customer orders should include the expected customer order sizes and customer order frequencies for each individual article in each planning period. This information can be obtained by analyzing the historical transaction data of the distribution center.

**Decision quantities.** Decisions about the following issues must be made. The list ranges from strategic or long-term decisions to tactical decisions.

- Selection of storage equipment, handling equipment, and storage policies.
- Physical layout of the storage accommodation with the size and location of the warehousing systems.
- Assignment of storage conditions to the different parts of the storage accommodation such as storage rooms.
- Number of planning periods per year.
- Determination for each article of the required storage space for each warehousing system and for each planning period.
- Allocation of storage space for each slot for each warehousing system and for each planning period.
- Assignment of articles to slots for each warehousing system and for each planning period.

**Constraints.** The following constraints must be met when the management makes decisions by assigning values to the decision quantities.

- The layout should not violate the dimensions of the building.
- The combined storage space of the slots should not violate the total available storage space of the storage accommodation.
- The allocated storage space for each article must be greater or equal to the minimum stock level of that article.
- The keeping quality loss of the articles may not exceed more than a predefined level agreed in the distribution chain.
- The time needed for handling of a customer order must not violate the agreed delivery time of the customer order.

**Objectives.** Given an instance of the problem, find values for the described decision quantities such that the sum of the keeping quality loss cost, the investment cost and the handling cost is minimal, and that the constraints are met.

The layout decisions determine the overall storage capacity of the distribution center. The choice of storage equipment influences the degree of storage space utilization and the expected total handling time. An assignment of articles to slots for each planning period is called a *slot plan*. The different storage conditions in the storage accommodation play a significant role in the assignment of the articles to slots because of the effect on keeping quality. Assigning articles to the same slots during the year reduces the training and search time for the personnel. The handling time is an indicator for the total handling cost because personnel cost outweighs equipment cost in warehouses with low levels of mechanization.

### 1.3 Towards solving the problem

There is a need for support of the management that deals with the presented problem. We state that this problem is relatively unstructured, based on two characteristics described by Savelsebergh [1988]. First, solving the problem is complex because the numerous objectives and constraints are difficult to quantify. In such a case, it is practically impossible to formulate a model that exactly mirrors the real-life situation. Secondly, the process required to find a feasible solution that is acceptable to the management cannot be completely specified in advance. We decided to facilitate the problem solving task of the management of the distribution center by introducing a *decision support system*.

#### 1.3.1 Decision support system

According to the definition given by Sprague & Watson [1986], a decision support system is an interactive computer-based system, which helps decision makers utilize data and models to solve unstructured problems. We
1.3. Towards solving the problem

consider the manager that deals with layout and operations management in the distribution center as the user of the decision support system. We assumed that this user is a trained professional, knowledgeable about the problem but not necessarily familiar with the techniques of operations research and information technology. The solutions to problems based on models alone cannot be used directly for decision making because models are never a perfect representation of reality. The decision support system should profit from the expertise of the user and the power and precision of the operations research models and methods. A decision support system can act as an assistant and as an advisor to the user. Models that are focused on evaluating decisions can assist the user and models that generate decisions can advise the user.

The decision support system that we developed tried to achieve the following three design goals, defined by Savelsbergh [1988] for decision support systems.

- Combination of operations research models and methods with data access and retrieval. The models will help the user to generate and evaluate alternative decisions. The quantitative methods depend on data of the problem instance.
- Flexibility and adaptability. The system must be able to accommodate changes in the problem situation and the solution process.
- Ease of Use. To achieve interaction between the system and the user, the system must be ‘user friendly’, so that the user can concentrate on the problem. A difficult to use system will probably remain unused, despite of the potential value.

The developed decision support system is called ADEPT. The acronym ADEPT stands for ATO-DLO Distribution Expert for Perishables in Transit. The system can complement existing standard warehouse management systems (WMS).

1.3.2 Hierarchical planning

We propose a hierarchical structure for the different, functionally related decisions that must be made to solve the problem. The structure divides the decisions into nested or hierarchical levels. The user can iterate between the different decision levels. The main advantage of such a solution approach is a reduction of the decision space on the separate levels. We can find several solution strategies based on hierarchical planning for the design of warehouses in the literature.
Park & Webster [1989] developed a detailed nine step procedure to design a three dimensional warehouse system. By varying the input data, the procedure generates alternative designs with analytical models. The alternatives are evaluated on investment and operations cost. The three stage hierarchical structure of Yoon & Sharp [1995] for the design of an order pick system follows basically the same procedure as Park & Webster [1989] but focuses on satisfying the requirements. Johnson & Lofgren [1994] evaluate the design alternatives for the order pick process with simulation models. The order pick process is decomposed in four subsystems which can be studied separately.

The design procedure of Rosenblatt & Roll [1984] tries to minimize operations cost and the rejection of shipments to the warehouse using simulation studies. On three different levels the user has to select the storage capacity, the storage policy and the type of layout. Larson, March & Kusiak [1997] let the user make these three decisions in the reverse order. Finally, Gray, Karmarkar & Seidmann [1992] developed a hierarchical planning procedure for an order consolidating warehouse that consists of the decision levels facility design and equipment selection, item allocation, and operating policy.

The overview demonstrates that there are several ways to decompose the problem, but none of the described procedures take into account the influence of the layout and operations management on the keeping quality loss of perishables. Storage conditions have an effect on the keeping quality loss and therefore restrict the possible assignments of articles to slots. The seasonality of supply and demand of perishables changes the stock levels of the articles during the year and therefore the storage space requirements. To our knowledge, the effect of seasonality on the expected handling time and the storage capacity utilization has not been studied yet. The general objective of our research is to develop models and solution strategies that support the decisions about layout and operations management for distribution centers for perishables. With these models and solution strategies the user can generate assignments of articles to locations and allocations of storage space, and evaluate the keeping quality loss, the expected handling time, and the storage capacity utilization of the proposed solutions.

For our solution strategy, we have chosen for a hierarchical decomposition into three decision levels that resembles the structure of Gray, Karmarkar & Seidmann [1992]. The decisions at the three levels are not very tightly
coupled. The relations between the levels are shown in Figure 1.1.

**Equipment and layout.** At the first level the user determines the required storage space for all the articles in the assortment in each planning period. The user also chooses the storage and handling equipment in the distribution center, and allocates storage capacity to storage rooms and warehousing systems. The majority of the decision at this level are strategic or long-term decisions. Even in an existing building, the number of choices at this level is enormous. The user has to find a balance between investment cost and storage capacity. High levels of storage capacity utilization often increase the expected handling time.

**Slot planning.** The main goal at the second level is to reduce the keeping quality loss due to inadequate storage conditions. Articles which require similar storage conditions are assigned to slots in the same storage room because with large assortments it is impractical to give each article its own storage room or each slot its own storage conditions. The tactical decisions at this level depend on the selected number of planning periods. More
planning periods reduce the effects of assortment changes and fluctuating stock levels on the storage space utilization. The slot plan also determines the expected handling time or work load.

**Operations management.** Operational decisions about the organization of the day-to-day work in the distribution center are made at the third level. The policies on how to act when a stock level of an article exceeds the storage space of an assigned slot are implemented as decision rules. With simulation models that incorporate these decision rules the behavior of the proposed solution can be studied.

The problems associated with the decision level ‘slot planning’ have several links with the decision levels ‘equipment and layout’ and ‘operations management’. The available storage capacity and the handling times are influenced by the layout and the equipment. The decisions about the number and the size of the storage rooms and warehousing systems have to be determined at the decision level ‘equipment and layout’.

### 1.3.3 Implementation of ADEPT

The three decision levels of the proposed hierarchical planning approach are implemented as separate modules in the decision support system ADEPT. Each module contains the data and the models needed to generate and/or evaluate the decisions at that decision level. During the solution process the user can iterate between the various decision levels to improve the solution quality. The output of the last visited level can be used as input for the next level. The user can also interact with the solution process by identifying interesting alternatives, and steering towards superior and practical layout and operations management plans. The alternative plans can be fine tuned with detailed simulation experiments to examine effects that have been overlooked or discarded in the models. The solution process eventually results in a solution with reduced keeping quality loss and lower investment and handling cost when compared with the initial solution.

A user-friendly implementation makes the frequent revisions in the plan an easy and straightforward job. The stand-alone system was built on a personal computer with the fourth generation language FoxPro 2.6 which includes a database management system. ADEPT has facilities to query the transactions database of the distribution center. We achieved interaction between the system and the user by making use of a graphical user interface. Histograms and other graphical presentations of the data can
1.4. Thesis outline

give a better insight into the daily fluctuations in the stock levels and the associated expected handling time. A drawing board for the layout makes the system truly visual interactive. At all levels of the solution process the user is able to view and edit the current plan. For example it is possible to view and edit the physical locations of the racks in a map of the layout of the distribution center.

1.4 Thesis outline

In our research we concentrate on the problem of the slot planning of distribution centers for perishables. The slot planning problem is linked with the decisions concerning the layout and equipment on one hand and the decisions dealing with the operations management on the other hand. Chapter 2 explains the specific problems associated with the distribution of perishables. The problem definition and the different decomposition strategies for slot planning are stated in Chapter 3. The different local search techniques that we used to solve the slot planning problem are introduced in Chapter 4. Chapter 5 deals with the effects of the layout and the slot plan on keeping quality loss. A simulation model is described that examines the advantages of a slot plan for the daily operations. The applied solution approach for the storage space allocation in the different warehousing systems is described in Chapter 6. In Chapter 7, we explain the effects of assignment of articles to locations on the expected handling time and on the storage space utilization. We illustrate the proposed solution strategies with a case example of a distribution center for vegetables and fruits in the Netherlands.
In this chapter, we introduce the specific problems of the distribution of perishables. After defining what product categories we consider to be perishable and describing their specific properties, we elaborate further on the production and distribution process. Thereafter, the consequences for the distribution chain of perishability are described, and the special considerations for perishables in distribution centers are explained. We conclude the chapter with the description of case examples of distribution centers for perishables.

2.1 Definition and properties of agricultural perishables

A large part of the fast moving consumer goods constitutes of perishables. In this thesis, we define perishables as stock items with a limited lifetime as reviewed by Nahmias [1982] and Raafat [1991]. Typical examples are food, photographic film and pharmaceuticals. Thereby, we restrict ourselves to perishables that are stockable, hence the perishable assets described by Weatherford & Bodily [1992] such as airline seats fall outside the scope of this research. Government regulations or the specific nature of the item can limit the lifetime to a fixed period. Blood and newspapers are examples of perishables with a fixed lifetime. The random lifetime of
perishables such as food is caused by unpredictable physical, chemical and biological processes that reduce the lifetime.

The past twenty years, the research on perishable inventory has concentrated mainly on bloodbanks instead of more interesting fields from a business point of view like the distribution of food. According to Nahmias [1982], the availability of data to the academic researchers has probably played a major role in the selection of the research topic for perishables. With the introduction of new technologies such as time-temperature monitors as described by Taoukis & Labuza [1989], more data have become available that are specific to perishables. In this research we focus on the distribution of fresh foodstuffs to fill the gap that is left by other researchers.

Fresh foodstuffs are usually from an agricultural origin, and consist of biological material. In the agribusiness, a product is a variety of for instance a vegetable with well defined properties like colour, taste and overall appearance. The diversity of marketed fresh foodstuffs in type, quality, and lifetime is enormous and is still increasing due to changing consumer preferences, globalization of the markets and technological innovations. Only a limited number of foodstuffs can be treated as normal commodities because of their relatively uniform quality and long lifetime. Examples are cereals and coffee. Meffert [1990] stresses that all other fresh foodstuffs have to be given special attention in the distribution chain from producer to consumer in order to deal with their specific properties such as aging and breakdown.

Fu & Labuza [1993] define keeping quality or shelf life as the period of time until which a product becomes unacceptable. According to Shewfelt [1990], the attribute that limits the product acceptance is the first attribute that becomes unacceptable, given the circumstances. The limiting attribute can also be predefined. An example is firmness of tomatoes in the study of Polderdijk, Tijskens, Robbers & Van der Valk [1994]. The keeping quality loss of food is based on biological processes which remain active after the harvest. Floros [1993] and Labuza [1982] mention a range of processes that influence keeping quality of agricultural products. The static keeping quality is the average number of days that a product is ‘fit for use’ if it is kept under the same storage conditions. Storage conditions which maximize the keeping quality result in minimal keeping quality loss.

The following items are examples of factors that influence the keeping quality; see also Ryall & Lipton [1979].
2.1. Definition and properties

- Initial keeping quality. The initial quality of an agricultural product varies because of differences in the used cultivars, the production method, the season and the production region.

- Storage time. Short storage times reduce the keeping quality loss.

- Temperature. Each product has its own optimal temperature. Many agricultural products that now or in the past originated from tropical or subtropical climates are susceptible to low temperature breakdown.

- Handling. Some perishables are very susceptible to shock and vibration. Suitable packaging can reduce the effects of shock and vibration during handling. Handling has in general a negative effect on keeping quality.

- Accessibility. The storage and handling equipment available at a storage room determine the accessibility of that storage room for handling. Low accessibility of a storage room corresponds with a large number of moves that are necessary to store and retrieve a product in that storage room, and results therefore in higher amounts of handling.

- Ethylene. The gas ethylene, a plant hormone for ripening, plays a role in the interaction between products. High concentrations of ethylene accelerate the ripening and decay of a large number of products. According to Abeles, Morgan & Salveit [1992], products that produce ethylene are susceptible to ethylene at the same time. The effect of the hormone ethylene can be avoided by spatially separating the products or reduced by ventilation of the storage room. The production of ethylene as well as the influence of this product interaction on the keeping quality are dependent on the temperature.

- Odors. Products like onions and garlic produce odors that are adsorbed by fruits such as melons. The packaging of the products can prevent keeping quality loss by odors. Milk is normally very susceptible to the odor of garlic, but milk in cartons does not pose any problems.

- Utilization. High levels of utilization reduce the free space in a storage room. The larger the free space in the storage room, the lower the concentration of odors and hormones.

- Ventilation. High ventilation rates reduce the effects of odors and hormones by increasing the effective volume in a storage room but increase the energy requirements.
Table 2.1: Characteristics of 60 quality change groups. The column 'T<sub>opt</sub>' denotes the optimal storage temperature, 'Prd' the ethylene production, and 'Sns' the ethylene sensitivity. For ethylene, the production and sensitivity are classified with 'H' for 'high', 'L' for 'low', and '-' for the absence of an effect. Data compiled by J.J. Polderdijk of ATO-DLO.

| Name quality change group | T<sub>opt</sub> (<°C>) | Ethylene | | | Name quality change group | T<sub>opt</sub> (<°C>) | Ethylene | | |
|---------------------------|----------------------|---------|---------|---------------------------|----------------------|---------|---------|
| Alfalfa                   | 0                    | -       | -       | Lettuce                  | 0                    | -       | H       |
| Asparagus                 | 0                    | -       | L       | Mango                    | 10                   | L       | H       |
| Babaco                    | 5                    | H       | H       | Melon                    | 10                   | L       | L       |
| Banana                    | 14                   | L       | H       | Mushroom                 | 0                    | -       | -       |
| Beetroot                  | 4                    | -       | -       | Paksoy                   | 0                    | -       | H       |
| Brasil                    | 4                    | -       | L       | Parsley                  | 0                    | -       | H       |
| Broccoli                  | 0                    | -       | H       | Passion fruit            | 6                    | H       | H       |
| Cabbage                   | 0                    | -       | H       | Pineapple                | 9                    | L       | -       |
| Carambola                 | 5                    | -       | -       | Plum                     | 1                    | L       | H       |
| Carrot                    | 0                    | -       | L       | Pomegranate              | 1                    | -       | -       |
| Cauliflower               | 0                    | -       | H       | Potato                   | 9                    | -       | L       |
| Cherry                    | 0                    | -       | -       | Quave                    | 8                    | L       | L       |
| Clementine                | 2                    | -       | L       | Raspberry                | 0                    | -       | -       |
| Corn                      | 5                    | -       | -       | Red bell pepper          | 8                    | L       | -       |
| Cranberry                 | 3                    | -       | L       | Rosemary                 | 0                    | -       | H       |
| Cucumber                  | 13                   | -       | H       | Rhubarb                  | 0                    | -       | -       |
| Curly kail                | 0                    | -       | L       | Sappodillo               | 10                   | H       | H       |
| Daikon                    | 0                    | -       | H       | Satsuma                  | 4                    | -       | L       |
| Egg plant                 | 8                    | -       | H       | Savory                   | 0                    | -       | H       |
| Elstar                    | 3                    | H       | -       | Scarlet runner           | 8                    | -       | L       |
| Endive                    | 0                    | -       | L       | Sharonfruit              | 8                    | L       | L       |
| Fig                       | 0                    | L       | -       | Spinach                  | 0                    | -       | H       |
| Garlic                    | 0                    | -       | -       | Sunkist                  | 5                    | -       | L       |
| Gherkin                   | 12                   | -       | H       | Strawberry               | 0                    | -       | -       |
| Golden delicious          | 1                    | H       | -       | Tamarillo                | 12                   | L       | L       |
| Green pepper              | 10                   | -       | L       | Taugeh                   | 0                    | -       | L       |
| Haricoverts               | 8                    | -       | L       | Tomato                   | 10                   | H       | L       |
| Jonagold                  | 2                    | H       | -       | Valencia late            | 2                    | -       | L       |
| Leek                      | 0                    | -       | L       | White grapefruit         | 7                    | -       | L       |
| Lemon                     | 12                   | -       | L       | Yellow melon             | 12                   | L       | H       |
2.2. Production and distribution

A quality change group consists of products with the same quality change characteristics, such as optimal temperature and production of ethylene. Examples of quality change groups are listed in Table 2.1. The quality change models used in this research describe for each quality change group the static keeping quality under specific storage conditions. Some of the mentioned factors can be controlled in the storage rooms of a distribution center. Each storage room has a specific temperature that is a compromise between the optimal temperatures of the quality change groups that are assigned to that storage room.

2.2 Production and distribution of agricultural perishables

Agricultural production is usually carried out in relatively small, family owned enterprises. Most production processes are soil bound and result in products with non-uniform keeping quality at the harvest. The number of different articles based on these products increases rapidly after the harvest due to activities like packaging and grading. The supply depends on the seasons. The consequence of these seasonal fluctuations is that products are not available during the entire year. The production pattern and the keeping quality are also dependent on the production region. Products from the southern hemisphere can substitute products from Europe during winter time and vice versa. In this way markets are supplied with fresh products during the entire year.

Customers determine more and more the type, quality, price, and availability of the articles. Customer service is becoming a very important item in the distribution chain, since the market for perishables has developed from a sellers market to a buyers market. The distribution chain of perishables is changing from a market ‘push’-situation to a ‘pull’-situation. According to Porter [1985], these trends are common for all consumer products. For perishables there is a mismatch between the long, seasonal production pattern and the short, day-to-day fluctuating consumption pattern. This results in a mismatch between supply and demand in the distribution chain. We distinguish between three systems to match the market push of the sellers and the market pull of the buyers.

- Forwards. The seller and the buyer draw up a contract that states type of product, quantity, quality, price, and time of delivery. In this way the buyer gets exactly what he wants but the system is not flexible. Adjustments in quantity, price, and time of delivery are difficult to handle.
- **Futures.** Standardized forwards that can be bought and sold on an open market like the Chicago Board of Trade are called futures. The possibility of trading increases the flexibility because contracts can be swapped for contracts with a more suitable quantity and time of delivery. The price forming is transparent. However, futures are only useful for products with relatively uniform quality that are bought in relatively large quantities.

- **Auction.** A place were sellers and buyers meet to negotiate openly about the price of the products on the market at that time is called an auction. The products never become the property of the auction. Rules about standardization of packaging and grading of the products create larger units that speed up the negotiation process. Buyers do not have a guarantee that the required product is available in the right quantity with the right quality. The sellers are on the other hand never certain of enough buyers for their products.

The effect of seasons and the type of contracts on the pricing is discussed by Desai [1996]. A new system that combines the advantages of futures and auctions for vegetables and fruits is proposed by Broekmeulen [1996a].

The different, and largely independent, actors at the different stages in the distribution chain often try to minimize the distribution cost through economies of scale. The producers are often organized in cooperatives. The auction is normally a cooperative of the producers. The auction system protects the small producers against the power of the buyers. A wholesaler usually buys goods on order instead of selling goods 'from the shelf' to the retailers. The retailers enlarge their buying power in large combinations. In an *integrated distribution chain* that is not controlled or owned by one enterprise, the margin improvement has to be distributed among the participating actors according to their controlling power in the distribution chain.

Hoekstra & Romme [1987] define the point in the distribution chain where the market push meets the market pull as the *decoupling point*. At a decoupling point the product can be buffered to reduce the mismatch between supply and demand. A buffer in the distribution chain of perishables has little use to close the gap between supply and demand because of the limited and random lifetime of the perishables. Shorter throughput times for the distribution process minimize the keeping quality loss, but the varying initial keeping quality and the varying conditions during the distribution process still result in a varying degree of keeping quality loss.
2.3 Integral quality control

Lack of harmonization in the distribution chains because of the independent actors at the different stages in the chain reduces the availability of perishables as consumer articles that meet a demand. The mismatch results in unnecessary cost, activities and keeping quality loss.

2.3 Integral quality control in distribution chains

Because of the specific storage conditions needed for each product and the mutual interference with other products, a distribution chain should be set-up in such a way that these special requirements are met during an as large as possible fraction of the supply lead time. Ideally, the mixing of the different products in the distribution chain has to take place in the last stages of the distribution process. The resulting compromise in the conditions for a mixed load in the last stages of transport and storage place a great burden on the keeping quality of the handled products. Therefore, the decoupling point for perishables must be located as close as possible to the producer. This is because the control of the distribution process gets complicated further down in the chain due to the divergence in product types and the difficulty to give each product the necessary specific storage conditions. This concept is promoted by Broekmeulen, Hoogerwerf, Simons & Reinders [1992] for the general situation, and further elaborated for mushrooms by Broekmeulen & Simons [1995]. Commodities such as cereals and potatoes are already stored by the producer. The distribution center of the wholesaler is the most practical location for the decoupling point in distribution chains that use the auction, since the wholesaler is the first large enterprise in the chain that owns the products. The flexibility in the distribution chain must be created in this distribution center.

The need for coordination as described by Slats, Bhola, Evers & Dijkhuizen [1995] and Thomas & Griffin [1996] is well understood in the distribution chain of agricultural perishables. At this moment there are only a few of such vertically coordinated supply chains operational. Examples are the supply of bananas and mushrooms. In these supply chains, one ‘dominant’ enterprise controls and owns the complete distribution chain of a perishable. Recent reorganizations of the supply chain of vegetables and fruits in the Netherlands can have a large impact on the coordination and the influence of the market, according to Van de Vorst & Simons [1996].

A correct and fast information flow between the actors is a requirement in the coordination of the distribution chain of perishables. Uniform article
coding and barcodes or radio frequency labeling are essential to achieve the necessary quality of the information and speed of the information flow. The large variation in product types, packaging, and quality have made uniform article coding still impractical for agricultural products. The short lifetime and the fast handling of the articles hinder the separate introduction of barcodes in the distribution center. The storage time is too short to give all incoming articles a detailed internal code or sticker. Personnel has to check by visual inspection that the correct article is handled. Wrong quality or packaging frequently result into mistakes.

2.4 Perishables in a distribution center

A distribution center performs activities such as storage, conditioning, accumulation and shipment. These activities need resources such as buildings, equipment, and personnel. Specific operating policies applied to a set of resources completes a warehousing system.

Figure 2.1: Types of article flows, the associated activities and the storage areas in a distribution center.
2.4. Perishables in a distribution center

2.4.1 Activities

The assortment of a distribution center consists of articles that are bought from various suppliers in the previous stages of the distribution chain. The customer orders are shipped to the next stage in the distribution chain in one or more consignments. The combination of one or more customer orders in a single consignment is called groupage. The different types of article flows from supplier to customer, and the associated activities are visualized in Figure 2.1. The primary function of a distribution center is groupage and shipment of customer orders.

With direct delivery from supplier to customer, the articles skip the distribution center. This option is interesting for large shipments of only a few articles. In all other cases, the articles enter the distribution center. All articles that are received in the distribution center are subject to inspection and quality control. Articles are received and shipped on pallets through dock boards or doors. A unit load is equal to a full pallet and carries a predefined number of article units that is specific for each article. Close to the dock boards there are two areas to temporary store articles: a receiving area for the incoming articles and a staging area for the outgoing articles. Each major customer has a reserved space or shipping area in the staging area for the pallets with articles ready for transport. In the shipping area, customer orders for more than one item of the assortment or for an article that has to be collected from several locations are accumulated before shipment to the customer. By cross docking, articles that are received and shipped the same day, and are ordered in unit loads, are collected directly from the receiving area into the staging area. The activity putaway is the movement of the articles from the receiving area to the slots in the storage area. By order picking the articles are collected from the storage area, and moved to the staging area. Picking personnel or pickers move the articles in the distribution center.

In this research, we consider two warehousing systems in the storage area: forward pick storage and reserve storage. The slots in the forward pick area are more easily accessible than the slots in the reserve area. Therefore, forward pick slots are the primary location for storage of an article and reserve slots are mainly used for bulk storage. Forward pick storage consists of forward pick slots at forward pick locations and reserve storage consists of reserve slots at reserve locations. Reserve picking stores and retrieves articles in unit loads from the reserve storage. Forward picking collects articles in less than unit loads from the forward pick storage. Inter-
nal replenishment moves the articles from reserve storage to forward pick slots. The activities cross docking, reserve picking and forward picking can all be involved in accumulating a customer order. The warehousing system that is preferred to collect a customer order depends on the order size and the unit load size for that specific article. Customer orders are split into handling requests for unit loads and for article units. Multiples of a unit load in a customer order are translated into multiple handling requests for reserve pickers.

2.4.2 Resources

The management has to make decisions about the available resources for the distribution center. These decisions are generated and/or evaluated at the decision level ‘layout and equipment’ of ADEPT.

Figure 2.2: A typical layout of a distribution center for perishables.

Building. A typical distribution center for perishables consists of a insulated building with an air-conditioned storage accommodation and several storage rooms as outlined by Sims [1994]. A typical layout is shown in Figure 2.2. An area in the storage accommodation with specific storage conditions is called a zone. The receiving and staging areas are not suit-
able for the storage of the articles for more than one day because of the absence of specialized conditions for each individual article and the mutual interference of the articles. Articles which have to stay in the distribution center for more than a day, or which are extremely susceptible to keeping quality loss, are put in storage in one of the zones. The slots in the storage area are not related to the customer but are chosen to meet the specific storage conditions and handling requirements of the articles. The \textit{i/o point} is a fixed point close to the dock board where each handling request or \textit{pick cycle} is considered to start and finish.

Figure 2.3: Cross section AB of a typical distribution center for perishables with slots in a rack with shelves.

\textbf{Equipment}. Articles can be stored in storage equipment such as racks and shelves. The type of packaging of an article has a strong influence on the possible storage equipment for that article. The storage equipment is arranged in \textit{aisles}. A typical rack is shown in Figure 2.3. Pallets on the upper levels of the racks can only be accessed with a forklift truck. Small package sizes such as trays and boxes cannot be handled easily with a forklift truck. Therefore, forward pickers use carts to handle article units and reserve pickers use forklift trucks to handle unit loads. Reserve pickers replenish the forward pick slots. A replenishment of less than a
unit load normally takes a considerable amount of time since the article units have to be restacked in the forward pick slot. Another option is that the unit load in the reserve storage is already split into loads that fit in the assigned forward pick slot by using additional pallet boards. The unit load for storage in the reserve storage is such cases received as a stack of smaller pallets.

The type of handling equipment and the degree of mechanization determine the differences in handling methods. The dimensions in the layout such as the maximum height of a rack and the minimum width of an aisle defines the possible handling equipment and vice versa. The chosen handling equipment and the degree of mechanization are a few of the factors that influence the maximum handling capacity or throughput of the distribution center and the productivity of the personnel. Modern equipment such as automatic carousels and automated sorters are often dedicated to the type of transport packaging and require specific storage equipment and high investments. The combination of racks, carts, and forklift trucks remains a flexible, low cost handling system with the disadvantage of relatively low throughput per picker. Because of the relatively low added value of perishables we assume the continuing use of carts and forklift trucks in the near future.

Personnel needs training to know what a specific article looks like, and whether the article picked for shipment is the same as is ordered by the customer. An administrative system such as a warehouse management system (WMS) can support the day-to-day operations by giving storage and retrieval advice to the pickers and generating pick lists. A pick list guides the pickers in the right sequence to the slots where the ordered articles must be present. A locator function records the storage to and retrievals from each slot real-time in the WMS. The records of the actual stock levels become inaccurate when the WMS cannot keep up with the pace of the physical transactions in the distribution center. This unwanted situation often occurs with fast moving consumers goods such as perishables. Logging all activities concerning storage and retrieval real-time in the case of perishables requires high investments in information technology and requires that all articles are labeled with a barcode or a radio frequency tag. An alternative system with dedicated primary locations for each article is simpler to control and relatively easy to learn for the pickers. Searching for the secondary location of the articles can take a considerable amount of time compared with an accurate WMS.
2.4.3 Operations management

The operations management of a distribution center for perishables includes decisions about inventory management and operating policies. The effects of these decisions are investigated on the decision level 'operations management' of the decision support system ADEPT.

Inventory management. Decisions about stock levels and external stock replenishment based on forecasts of supply and demand is impractical because of the mismatch between the long, seasonal production pattern and the short, day-to-day fluctuating consumption pattern, and of the lack of harmonization in the distribution chains. The trend of the seasonal changes is known or can be forecasted, but the daily fluctuations are enormous. Stock that is kept to deal with unpredictable demand is called safety stock. Cycle stock is stock kept to deal with unpredictable lead times of suppliers. According to Silver, Pyke & Peterson [1998], safety stock and cycle stock are both additions to forecasted stock levels to avoid 'out-of-stock' situations. Safety and/or cycle stock are not useful for perishables. because the additional keeping quality loss of the perishables in the safety and/or cycle stock is unacceptable for the customers. Therefore, most articles are bought on order so that the customer of the articles is already known when they enter the distribution center. The customer lead time is kept as short as possible. The time in stock or storage time is rather half a day than a few weeks. The number of times that the average storage time fits in a year is denoted by the turnover rate. For a warehouse, Rosenblatt & Roll [1988] define the storage service level as the percentage of putaway requests that can be satisfied during a planning period because of available storage space. The allocated storage space in the storage area is related to the storage service level and the average storage time of an article in a planning period. The stock of an article can be located in more than one warehousing system.

When the probability of the occurrence of an article on a customer order is affected by the occurrence of other articles on that order, we have demand dependency. Models that include demand dependency are discussed by Frazelle [1989] and Kim [1993]. For fast moving consumer goods such as perishables that are ordered on a daily basis, demand dependency plays no important role since most articles are requested in every customer order.

Operating policies. The operating policies and the accuracy of the WMS influence the amount of keeping quality loss and the efficiency of the operations. The personnel in the warehouse needs storage and retrieval
advice in order to subsequently reduce the search time for available slots, to store the article and to find the slot where the ordered article can be retrieved. This advice depends mainly on the storage and/or retrieval policy of the distribution center. We only consider the First In First Out (FIFO) retrieval policy due to the perishable nature of the articles. Alternative retrieval policies such as Last In First Out (LIFO) result in longer durations of stay for the articles and thus in too much keeping quality losses.


- **Fixed slot policy.** Storage or retrieval involves dedicated, assigned slots. If the storage capacity of a slot is insufficient with regard to the number of article units that have to be stored, the purchase could be temporarily put somewhere else and may even get ‘lost’

- **Closest available slot policy.** If the assigned slot is occupied, the most closest available and suitable slot is taken.

- **Random slot policy.** Each storage request gets the first available slot in the warehouse.

All above described policies need accurate information about the available slots for an article and the contents of the occupied slots. If a picker cannot find an article at the dedicated slot or at a slot indicated by the WMS, all alternative slots have to be checked before the article is reported out-of-stock. Note that in the absence of a real-time locator, storage and retrieval from alternative slots require additional time for searching.

In a distribution center for perishables, all articles that are kept in stock need a fixed slot or location due to the absence of a uniform article coding and the special conditions needed by perishables. A policy with random slots has a better space utilization than a system with only fixed slots according to Francis, McGinnis & White [1992], because empty slots reserved for a specific article cannot be used by another article. In practice, the distribution center can work with a fixed slot policy for the primary locations in the forward pick storage and a closest available slot policy for the reserve storage. In the reserve storage of each zone, the articles can be assigned to shared slots for subsets of articles to improve the storage space utilization of the zones.

The user can change the slot plan for each planning period to improve the utilization of the storage capacity. Frequently changing the slot plan increases the time needed to search the articles and to train the pickers.
2.4. *Perishables in a distribution center*

Figure 2.4: An example of the usage of two racks by the articles A, \ldots, E during three planning periods without fixed article assignments to slots throughout the year.

A slot plan with fixed locations of each article throughout the year needs less handling time but requires more storage space. The problem resulting from fixed locations in the slot plan is illustrated for a rack in Figure 2.4. Note that the forward pick location of article C has to change in period three, because of the greater storage space allocation of article C and the continued occupation of the shelf by article D.

Possible order picking strategies are reviewed by Frazelle & Apple [1994]. The choice for a strategy depends on the number of pickers assigned to each customer order, the number of customer orders handled during each pick cycle and the method of accumulation of a customer order. In order to meet a certain customer lead time, it may be necessary to divide a large customer order over several pickers. Pickers assigned to the same customer order can work in different warehousing systems or aisles. More than one picker working in the same aisle can result in congestion when the pickers hinder each other during their work.

In a *picker-to-part* system, a pick cycle starts from the i/o point with an
empty cart or forklift truck, visits one or more slots, grabs the requested number of article units at the slots and returns to the i/o point with a partial or complete customer order. With batch picking each pick cycle collects for more than one customer order. The total volume of the customer orders that are combined in a batch may not exceed the capacity of the handling equipment of the picker. Remember that usually a unit load the largest volume is that a picker can handle in a pick cycle. A distribution center specialized in perishables usually handles relatively large orders of distribution centers of the large retail and supermarket chains, which makes batch picking impractical. A pick cycle for a single customer order results in a partial customer order if the complete customer order consists of more than one unit load, if more than one picker is assigned to the customer order, or if the stock is insufficient. Articles that are not in stock because they did not arrive in time in the distribution center have to be picked in another pick cycle later that day.

Figure 2.5: The routing of a single address pick cycle from the i/o point to a slot in one of the racks and back.

A pick cycle can be distinguished in a single-address and a multi-address pick cycle. Multi-address pick cycles, where a picker visits more than one location during a pick cycle, is among others discussed by Malmborg & Kr-
ishnakumar [1989] and De Koster, M.B.M., Van der Poort & Roodbergen [1997]. The routing of a single-address pick cycle is shown in Figure 2.5. The travel time to a slot depends on the speed of the equipment used for picking and the distance between the i/o point and the slot. The calculation of the travel time is relatively straightforward for a single-address pick cycle. The distance approximations needed for multi-address pick cycles are investigated by Hall [1993].

A multi-address pick cycle of a forward picker starts with an empty pallet and often results in a pallet with different articles in different packagings. Such a mixed pallet has to be stable for further transport. Heavy articles put on top make the pallet collapse. A box cannot be placed on a sack but a tray can be placed on a box. The forward picker has to sort or rearrange the stacking of the articles on the pallet during the pick cycle to keep the pallet stable. Correct stacking can also prevent damage to fragile products. Sorting of the articles during a pick cycle because of stacking problems has to be avoided. With a fixed pick route, the heavy articles in easily stackable packaging are placed at the beginning and sacks with light articles are placed at the end of the pick route. A possible strategy with variable pick routes is to assign to each rack or aisle only articles with the same packaging. When single-address pick cycles are used and/or more than one picker is assigned to a customer order, the articles have to be sorted when the complete customer order is accumulated.

2.5 Case examples of distribution centers for perishables

In this study we used real world data of distribution centers for perishables in the Netherlands. The following case example of a wholesaler of vegetables and fruits gives an indication of the real world problem sizes.

From a major wholesaler of vegetables and fruits in the Netherlands we received a complete and detailed dataset of a whole year concerning one of the distribution centers of the wholesaler. The data on the day-to-day stock levels was sometimes inaccurate since the installed administrative system could not keep up with the pace of the physical transactions in the distribution center. We describe this case example in more detail. The distribution center of the wholesaler is a 4000 m² facility with a height of 10 m with four cold stores of 1500 m³ each. The complete assortment lists more than 5000 articles divided into 180 quality change groups. On a single day not more than 750 different articles are in stock due to seasonal
variations in the assortment. The standard unit load is a pallet with a length of 1.2 m, width of 1.0 m, and an average height of 2 m. The distribution center handles about 400 pallets in and out of storage each day. Six days a week the distribution center buys vegetables and fruits on order at auctions and directly from growers for about 20 customers. These customers are large supermarket chains. Every sunday the distribution center is about empty.

Figure 2.6: The aggregated stock level of all the distributed articles of the wholesaler of vegetables and fruits during a year.

The current storage conditions in the warehouse of the wholesaler did not take the specific quality change properties of the assortment in consideration. Three of the four cold stores had a temperature of 1 °C, which made them effectively one single zone. Especially the product interactions were overlooked at higher zone temperatures.

Figure 2.6 shows the yearly fluctuations of the actual aggregated stock level of the distribution center. Note that the stock level has a dip near the end of the summer season and that the week before Christmas has the highest stock level. Figure 2.7 shows the variation of the stock level of eight quality change groups with respect to the seasons. The multiple peaks in most plots can be attributed to different production seasons in
2.5. Case examples

Figure 2.7: The stock level of eight quality change groups in the distribution center of the wholesaler of vegetables and fruits during a year.
Figure 2.8: The summed peak stock levels of all the quality change groups as a function of the number of planning periods in the dataset of the wholesaler of vegetables and fruits.

different parts of the world or to different production methods.

The peak stock level of an article in a planning period depends on the time of year and the length of the planning period. A longer planning period increases the chance that a particular peak stock level is included in that planning period. The summed daily stock levels of complementary articles, such as endive and strawberry, results in a lower peak stock level than the sum of the individual peak stock levels. The dip in stock from endive is compensated by the peak of strawberry, and vice versa. The size of that effect is shown in Figure 2.8, where we summed the peak stock levels of the quality change groups. We combined the stock levels of the articles that belong to the same quality change group because these articles have to be stored at the same location under identical storage conditions. The summed peak stock level decreases when we increase the number of planning periods. Even with 20 planning periods, the summed peak stock levels of the quality change groups is two times the summed daily stock levels of all the articles in the distribution center. We conclude
2.5. Case examples

Figure 2.9: The size of, and the changes in the weekly assortment of the wholesaler of vegetables and fruits during a year.

that the required storage capacity for an article or a quality change group is always higher than the average stock level but much lower than the peak stock level. If we assign complimentary articles to a single location, then the utilization of that location during the year can remain high during all planning periods.

The size of, and the changes in the weekly assortment of the wholesaler are shown in Figure 2.9. We observe three major changes in the assortment around week 51, week 18, and week 36, which coincide with the beginning of the following three seasons: early, Holland and late. The assortment in the Holland season lists mainly articles from Dutch growers. The corresponding reduced number of imported articles in the Holland season explains the lower number of articles in the assortment. The early and late season rely on imported articles. Christmas marks the change from the late to the early season.
Distribution of perishables
3

Slot planning

In this chapter we present a formal definition of the slot planning problem in a distribution center. In the slot plan, one has to decide where the articles are placed and how much space must be allocated to each article stored in the distribution center and for each planning period during the year. This slot plan directly influences the keeping quality of the articles, the handling time and the utilization of the storage space. Since slot planning is mainly concerned with the warehousing function of a distribution center, this problem is called the warehouse slot planning problem (WSPP). Thereafter, the warehouse slot planning problem for perishables (WS3P) is formulated as a special case of the WSPP. Next, we discuss aspects of the WSPP and the WS3P such as handling and quality change models. The problem analysis focuses on the problem size and complexity. We propose two basic strategies to solve both slot planning problems: handling and space oriented, which decomposes the problem into smaller, easier to handle subproblems.

3.1 The warehouse slot planning problem

We define the warehouse slot planning problem (WSPP) as a partitioning problem with constraints on the storage capacity. Before defining the
WSPP and the associated handling models, we give following the definitions of *slot data* and *slot plans*.

**Definition 3.1** (Slot data). An instance of the slot data is given by a 15-tuple $SD = (A, T, w, w_f, L, V, V_f, R, F, l_s, v, S, \Theta, \omega_h, \omega_u)$, with

- $A$, a set of articles $a$,
- $T$, a set of planning periods $t$,
- $w : A \times T \to \mathbb{N}_0^+$, a function that gives the minimum required total storage space for an article in a planning period,
- $w_f : A \times T \to \mathbb{N}_0^+$, a function that gives the minimum required forward pick storage space for an article in a planning period,
- $L$, a set of locations $l$,
- $V : L \to \mathbb{N}^+$, a function that gives the total storage capacity of a location,
- $V_f : L \to \mathbb{N}^+$, a function that gives the forward pick storage capacity of a location,
- $R$, a set of reserve slots,
- $F$, a set of forward pick slots,
- $l_s : R \cup F \to L$, a function that gives the location of a slot,
- $v : R \cup F \to \mathbb{N}^+$, a function that gives the storage capacity of a slot,
- $S \subseteq \mathcal{P}(R \cup F)$, a set of sets of slots $s \in R \cup F$ to which an article can be assigned,
- $\Theta : A \times T \times S \to \mathbb{N}_0^+$, a function that gives the expected handling time for an article occupying a set of slots in a planning period,
- $\omega_h \in \mathbb{N}_0^+$, a weightfactor for handling, and
- $\omega_u \in \mathbb{N}_0^+$, a weightfactor for storage capacity utilization.

The set of all instances of the slot data is denoted by $SD$. □

Handling time is expressed in seconds per year, and storage space and storage capacity are expressed in m$^3$. The weightfactor $\omega_h$ assigns a cost to each second of handling time and the weightfactor $\omega_u$ assigns a cost to each utilized location during a year.

Each article that is handled and stored in the distribution center needs a location and storage space. An article may occupy more than one slot in the distribution center. This results in the following definition of a slot plan.
3.1. The warehouse slot planning problem

Definition 3.2 (Slot plan). Given is an instance of slot data $SD = (A, T, w, w_f, L, V, V_f, R, F, l_a, v, S, \Theta, w_h, w_u) \in SD$, as described in Definition 3.1. A slot plan $\chi$ for $SD$ is a function $\chi : A \times T \rightarrow S$ which assigns each article in a planning period to a set of slots with a fixed location and a fixed storage capacity. The associated utilization of the locations is described by the function $y : L \rightarrow \{0, 1\}$. Here "1" corresponds to the situation that the location is used and otherwise the location is free, denoted by "0".

The size of the assortment is represented in the set of articles $A$. The decision about the number of planning periods can be made after analysis of the seasonality of the assortment. The minimum required storage space $w$ for an article depends on the required storage service level for an article and the average storage time of an article. The minimum required forward pick storage capacity $w_f$ is determined by the average customer order size and the required customer lead time for an article.

We restrict ourselves to two warehousing systems: reserve and forward pick storage. The decisions made at the decision level 'equipment and layout', introduced in Section 1.3.2, give the available storage equipment and the positions of this equipment in the warehouse. In this research, we denote each item of storage equipment with its position such that each section of a rack or shelving system corresponds with locations in the set $L$ and the associated storage capacities with $V$ and $V_f$. Therefore, each slot in a section of a rack has the same location. We assume that forward pick storage capacity can also be used as reserve storage capacity, but for each location the forward pick storage capacity $V_f$ is always less or equal to the total storage capacity $V$. The sets $R$ and $F$ describe possible configurations of reserve and forward pick slots at the available locations in the warehouse.

Forecasts or historical data on the customer orders are included in the handling function $\Theta$, because the expected handling time also depends on the customer orders that have to be handled in the warehouse. A distribution center may use a slot plan for several years if the yearly turnover remains constant over the years.

3.1.1 Handling models

The expected handling time $\Theta$ depends on the actual or forecasted customer orders, the layout and equipment of the warehouse and the slot
plan. We developed generalized handling models to determine the expected total handling time in a planning period for each assignment of an article to a combination of reserve and forward pick slots.

**Assumptions.** For the handling model at the decision level ‘slot planning’, the following assumptions were made with respect to the handling operations.

- *Activities.* The models include the following activities or *handling operations:* putaway, reserve picking, replenishment and forward picking. Other operations such as cross docking, cycle counting and inspection are assumed to be independent of the slot plan and are therefore not included in the handling model.

- *Pick strategy.* Each article of each customer order is handled separately out of storage with single-address pick cycles. This assumption of single-address pick cycles eliminates the possible effects of demand dependency. The effects of accumulating these handling requests for articles into complete customer orders are treated at the decision level ‘operations management’, introduced in Section 1.3.2.

- *Storage policy.* Handling operations are carried out to and from a set of fixed slots for each article.

- *Handling equipment.* Forward pickers and reserve pickers use different handling equipment. The applied handling equipment determines the speed of the handling operations, where the forward picking equipment is slower than the reserve picking equipment. Handling equipment can handle one unit load at a time.

- *Handling times.* The time to unload the handling equipment and receiving a new pick instruction is fixed for each pick cycle and is independent of the used handling equipment. The handling equipment and the location of the slot determine the time needed to search for and to stop at the slot, and the travel time from the i/o point to the slot and vice versa. The grab time at the slot depends on the number of the article units ordered and the type of handling equipment, but not on the type of slot. The time needed to place or extract a unit load is also called grab time. A schematic overview of the different work elements in a single-address pick cycle is shown in Figure 3.1.

- *Putaway.* Articles that are picked from the storage area have to be stored first. Putaway handles the articles in unit loads with reserve pick equipment.
3.1. The warehouse slot planning problem

- Reserve picking. An customer order for a multiple of unit loads is always handled by a reserve picker, and when possible collected from a reserve slot. This assumption differs from the assumption made in the handling model of Van den Berg [1996].

- Replenishment. Reserve pickers carry out internal replenishments from reserve storage to forward pick storage. A forward pick slot has a maximum storage capacity equal to the maximum unit load volume. Replenishment fills an empty forward pick slot up to the available storage capacity from reserve storage. The handling time for each replenishment request is fixed and depends only on the perimeter of the warehouse.

- Forward picking. A forward picker visits only forward pick slots and handles less than unit loads.

- Congestion. There is no congestion in the aisles. This remains true as long as there are only a few pickers present in each aisle.

Figure 3.1: A schematic overview of the different work elements that are included in the handling time of a single-address pick cycle.

Based on the above assumptions, we model the expected handling time $\Theta$ for an article in a planning period as the sum of the handling operations putaway, reserve picking, forward picking and replenishment.
**Definition 3.3** (Expected handling time). The expected handling time $\Theta$ for each article $a \in A$ in a slot plan $\chi$ during a planning period $t \in T$ is given by the sum of the individual handling operations putaway, reserve picking, forward picking and replenishment. Hence,

$$\Theta(a, t, \chi(a, t)) = [\Theta_{ptw}(a, t, \chi(a, t)) + \Theta_{rpk}(a, t, \chi(a, t))$$

$$+ \Theta_{fkp}(a, t, \chi(a, t)) + \Theta_{rpl}(a, t, \chi(a, t))] \quad (3.1)$$

where

- $\Theta_{ptw} : A \times T \times S \rightarrow \mathbb{R}_0^+$ gives the total time for putaway of an article in a set of slots in a planning period,
- $\Theta_{rpk} : A \times T \times S \rightarrow \mathbb{R}_0^+$ gives the total time for reserve picking of an article in a set of slots in a planning period,
- $\Theta_{fkp} : A \times T \times S \rightarrow \mathbb{R}_0^+$ gives the total time for forward picking of an article in a set of slots in a planning period, and
- $\Theta_{rpl} : A \times T \times S \rightarrow \mathbb{R}_0^+$ gives the total time for replenishment of an article in a set of slots in a planning period.

The function $\lfloor r \rfloor : \mathbb{R} \rightarrow \mathbb{N}$ gives the integer part of a real number $r$. □

The time needed for these handling operations depends on the characteristics of the customer orders, the handling equipment and the occupied slots.

**Customer orders.** All handling operations are based on the set of actual or expected customer orders. Because we restrict ourselves in the handling model to customer orders that are collected from the storage area, we omit customer orders that are handled ‘cross dock’ and customer orders that are shipped directly from supplier to customer from the set of customer orders.

The selection of handling operations for a customer order depends on the number of article units ordered and the number of articles in a unit load. The unit load is defined as follows.

**Definition 3.4** (Unit load). A unit load or unit load quantity of an article $a \in A$ is equal to the number of article units that fit on a pallet, i.e., $\beta : A \rightarrow \mathbb{N}^+$. The volume of the unit load may not exceed the maximum unit load volume $\omega_u \in \mathbb{N}^+$. □

In the distribution center the customer orders are changed in handling requests for single-address pick cycles. Since pickers are not able to handle
more than one unit load at the same time, we have to divide each customer order into one or more handling requests of one unit load or less. We define a handling request as a pick cycle for an article in a customer order and the total number of articles ordered of an article in a planning period as the order frequency. We use separate definitions for the number of handling requests and the order frequency for unit loads and article units, based on the number of articles units in a unit load.

**Definition 3.5** (Handling requests). The total number of handling requests \( \eta \) for an article \( a \in A \) in planning period \( t \in T \) is equal to

\[
\eta(a, t) = \eta_u(a, t) + \eta_a(a, t) ,
\]

where
- \( \eta_u : A \times T \to \mathbb{N}_0^+ \) gives the number of handling requests for unit loads of an article in a planning period, and
- \( \eta_a : A \times T \to \mathbb{N}_0^+ \) gives the number of handling requests for article units of an article in a planning period.

**Definition 3.6** (Order frequency). The total order frequency \( \gamma \) of article \( a \in A \) in planning period \( t \in T \) is the number of articles ordered in that planning period, hence

\[
\gamma(a, t) = \beta(a) \cdot \gamma_u(a, t) + \gamma_a(a, t) ,
\]

where
- \( \gamma_u : A \times T \to \mathbb{N}_0^+ \) gives the number of unit loads of an article in a planning period that are handled as unit loads, and
- \( \gamma_a : A \times T \to \mathbb{N}_0^+ \) gives the number of article units of an article in a planning period that are handled as article units.

The order frequency \( \gamma_u \) is expressed in unit loads per planning period and the order frequencies \( \gamma \) and \( \gamma_a \) are expressed in article units per planning period.

**Handling times.** The time spent on each individual handling operation is based on the time required for the separate work elements of a single-address pick cycle, as illustrated in Figure 3.1. We split the work elements for each pick cycle in two categories of handling times: cycle time and grab time. The cycle time, needed for subsequently getting the pick instruction, traveling, stopping at the slot and unloading at the i/o point, correlates with the number of handling requests. The cycle time consists of a fixed
and a variable part. The fixed cycle time includes getting the pick instruction or loading and unloading at the i/o point. The variable cycle time includes travel to and from the slot. The grab time is proportional to the order frequency.

**Definition 3.7** (Cycle time). The cycle time includes the following work elements. Travel to and from the slot, stopping at the slot and the loading/unloading at the i/o point. The cycle time differs for each type of slot and each type of handling equipment, i.e.,

- $\theta_{cr}: \mathcal{R} \rightarrow \mathbb{N}^+$ gives the cycle time to a reserve slot needed by a reserve picker,
- $\theta_{cf}: \mathcal{F} \rightarrow \mathbb{N}^+$ gives the cycle time to a forward pick slot needed by a reserve picker, and
- $\theta_{cf'}: \mathcal{F} \rightarrow \mathbb{N}^+$ gives the cycle time to a forward pick slot needed by a forward picker,

where $\mathcal{R}$ is the set of reserve slots and $\mathcal{F}$ is the set of forward pick slots. The travel distance to the slot depends on the location of the slot. 

The function $\theta_{cf}$ for the cycle time to a reserve slot by a forward picker was omitted since a forward picker never visits a reserve slot. Since we assumed that the handling equipment of a forward pickers is slower than that of a reserve picker, the cycle time for a reserve picker to a forward pick slot is always less than for forward picker to the same slot, hence $\theta_{cf} < \theta_{cf'}$. Therefore, unit loads are never picked from forward pick slots by a forward picker but by a reserve picker. In order to make better use of the forward pick storage, we assumed that unit loads are picked from reserve storage whenever possible. Even when the article occupies storage space in the forward pick storage. In this way, unnecessary time consuming replenishment operations are avoided.

A slot plan may assign an article in a particular planning period to more than one slot for each warehousing system. The total handling time correlates with the utilized storage capacity because each additional slot may increase the walking distances of the picking personnel. Therefore, we assume that articles are only assigned to more than one slot when these slots are also occupied most of the time during the planning period. This is the case when we try to minimize handling time and maximize storage capacity utilization at the same time. The average stock level of an article in a slot is usually 50 %. Higher average stock levels can be achieved when each storage request is preceded by a retrieval request for the same amount
of article units. Applying the FIFO rule together with a high utilization
of the slots makes that each slot is visited with the same frequency when
the storage capacities of slots are equal. With slots of different storage
capacities, the number of article units that are picked from a slot depends
on the capacity of the slot relatively to the total available storage capacity
of reserve or forward pick slots for that article.

First, we determine the total available storage capacity in a set of slots.

**Definition 3.8** (Set storage capacity). The total available storage capac-
ity $V_s$ in a set of slots $U \in S$ is equal to the sum of the storage capacities
$v(s)$ of the slots $s \in U$, i.e.,

$$V_s(U) = \sum_{s \in U} v(s).$$

The weighted cycle times for a slot plan can now be defined as follows.

**Definition 3.9** (Weighted cycle time). The weighted cycle time for a set
of slots $U \in S \subseteq \mathcal{P}(R \cup F)$ is equal to the average cycle time of all the
slots $s \in U$, weighted over the storage capacity of the slot $v(s) \in \mathbb{N}^+$, i.e.,

- $\bar{\theta}_{crr} : S \rightarrow \mathbb{R}^+$ gives the weighted cycle time to a set of reserve slots
  needed by a reserve picker, i.e.,

$$\bar{\theta}_{crr}(U) = \frac{\sum_{s \in U \cap R} \theta_{crr}(s)v(s)}{V_s(U \cap R)},$$ (3.4)

- $\bar{\theta}_{cfr} : S \rightarrow \mathbb{R}^+$ gives the weighted cycle time to a set of forward pick
  slots needed by a reserve picker, i.e.,

$$\bar{\theta}_{cfr}(U) = \frac{\sum_{s \in U \cap F} \theta_{cfr}(s)v(s)}{V_s(U \cap F)}, \text{ and}$$ (3.5)

- $\bar{\theta}_{cff} : S \rightarrow \mathbb{R}^+$ gives the weighted cycle time to a set of forward pick
  slots needed by forward picker, i.e.,

$$\bar{\theta}_{cff}(U) = \frac{\sum_{s \in U \cap F} \theta_{cff}(s)v(s)}{V_s(U \cap F)},$$ (3.6)

where $R$ is the set of reserve slots, $F$ is the set of forward pick slots,
and $V_s$ gives the total available storage capacity of a set of slots. For
the description of the other sets and functions, we refer to Definitions 3.1
and 3.7. □
Because we assumed that the grab time is independent of the type of slot, the grab time is also independent of the slotplan.

**Definition 3.10** (Grab time). The grab time depends on the type of load and the type of handling equipment, i.e.,

- $\theta_{gur} \in \mathbb{N}^+$ denotes the time needed by a reserve picker to place a unit load in or to extract a unit load from a slot,
- $\theta_{gar} \in \mathbb{N}^+$ denotes the grab time for an article unit of a reserve picker, and
- $\theta_{gaf} \in \mathbb{N}^+$ denotes the grab time for an article unit of a forward picker.

For a reserve picker, the grab time for an article unit is usually greater than or equal to the grab time for a unit load, hence $\theta_{gur} \leq \theta_{gar}$. This is caused by the type of equipment of a reserve picker since these are less suited for picking small items.

The time for a replenishment request is defined as follows.

**Definition 3.11** (Replenishment time). The replenishment time for replenishing at most one unit load is equal to $\bar{\theta}_{rpl} \in \mathbb{N}^+$.

We stated that the replenishment time mainly depends on the layout of the warehouse and more specific on the perimeter of the warehouse. Remember that the reserve picker that carries out a replenishment can only handle one unit load at a time and that the storage capacity of a forward pick slot is less or equal to the maximum unit load volume $\omega_u \in \mathbb{N}^+$. More storage space for an article in the forward pick area can be achieved by assigning the article to more than one slot.

**Assignment to warehousing systems.** The way that an article $a \in \mathcal{A}$ is assigned to the forward pick and/or reserve storage in a planning period $t \in \mathcal{T}$ determines which handling operations take place for storage and retrieval.

If the minimum required total storage space $w(a, t)$ equals 0, then the articles are not put in storage, but are handled ‘cross dock’, as is illustrated by Figure 2.1. Cross docking does not affect any of the components of the handling model, hence the required handling time $\Theta(a, t, \chi(a, t))$ equals 0. In all other cases when $w(a, t)$ is greater than 0 the required handling time depends on one of the following three situations.
3.1. The warehouse slot planning problem

![Diagram showing three situations for the assignment of articles to the reserve and/or forward pick storage.]

Figure 3.2: Three situations for the assignment of articles to the reserve and/or forward pick storage.

1. **Reserve storage only**, i.e., $\Theta = \Theta_{ro}$. If the total available forward pick storage capacity $V_x(\chi(a,t) \cap \mathcal{F})$ equals 0, this means that the article is stored and retrieved from reserve slots only. Then the handling functions are given by

$$
\Theta_{ptw}(a,t,\chi(a,t)) = \left(\bar{\theta}_{crr}(\chi(a,t)) + \theta_{gur}\right) \cdot \left[\frac{\gamma(a,t)}{\beta(a)}\right],
$$

$$
\Theta_{rpk}(a,t,\chi(a,t)) = \bar{\theta}_{crr}(\chi(a,t)) \cdot \eta(a,t) + \theta_{gur} \cdot \gamma_u(a,t)
$$

$$
+ \theta_{gar} \cdot \gamma_a(a,t),
$$

$$
\Theta_{fpk}(a,t,\chi(a,t)) = 0, \text{ and}
$$

$$
\Theta_{rpl}(a,t,\chi(a,t)) = 0,
$$

where $\text{ceil}(x)$ gives the first integer that is greater or equal than $x$.

2. **Forward pick storage only**, i.e., $\Theta = \Theta_{fo}$. If the total available reserve storage capacity $V_x(\chi(a,t) \cap \mathcal{R})$ equals 0, and the total available forward pick storage capacity can hold the minimum required total storage space, i.e., $w(a,t) \leq V_x(\chi(a,t) \cap \mathcal{F})$, the article is stored and retrieved from forward pick slots only. There is no reserve picking or replenishment possible. Hence, the handling functions are equal.
to

\[ \Theta_{ptw}(a, t, \chi(a, t)) = (\bar{\theta}_{cr}(\chi(a, t)) + \theta_{gur}) \cdot \left( \frac{\gamma(a, t)}{\beta(a)} \right) , \quad (3.11) \]

\[ \Theta_{rpk}(a, t, \chi(a, t)) = 0 , \quad (3.12) \]

\[ \Theta_{fpk}(a, t, \chi(a, t)) = \bar{\theta}_{cr}(\chi(a, t)) \cdot \eta(a, t) \]
\[ + \theta_{gaf} \cdot \gamma(a, t) , \quad \text{and} \]
\[ \Theta_{rpl}(a, t, \chi(a, t)) = 0 . \quad (3.14) \]

3. Reserve and forward pick storage, i.e., \( \Theta = \Theta_{rf} \). If the minimum required total storage space is greater than the total available forward pick storage capacity or \( \omega(a, t) > V_s(\chi(a, t) \cap F) \), and the total available forward pick storage capacity \( V_s(\chi(a, t) \cap F) \) is greater than 0, then forward pick slots must be replenished from reserve slots. Remember that we assumed picking of unit loads from reserve storage even when there is forward pick storage allocated to the article. Consequently, the handling functions in the situation of reserve and forward forward pick storage are given by

\[ \Theta_{ptw}(a, t, \chi(a, t)) = (\bar{\theta}_{cr}(\chi(a, t)) + \theta_{gur}) \cdot \left( \frac{\gamma(a, t)}{\beta(a)} \right) , \quad (3.15) \]

\[ \Theta_{rpk}(a, t, \chi(a, t)) = \bar{\theta}_{cr}(\chi(a, t)) \cdot \eta_u(a, t) \]
\[ + \theta_{gur} \cdot \gamma_u(a, t) , \quad (3.16) \]

\[ \Theta_{fpk}(a, t, \chi(a, t)) = \bar{\theta}_{cr}(\chi(a, t)) \cdot \eta_a(a, t) \]
\[ + \theta_{gaf} \cdot \gamma_a(a, t) , \quad \text{and} \]
\[ \Theta_{rpl}(a, t, \chi(a, t)) = \frac{\bar{\theta}_{rpl}}{V_s(\chi(a, t) \cap F)} \cdot \frac{\gamma_a(a, t)}{\beta(a)} . \quad (3.18) \]

Figure 3.2 shows the different article assignments with respect to the three situations. On the left we have illustrated the ‘reserve storage only’ situation which is interesting for articles that are requested mainly in unit loads. This situation occurs for articles with small number of article units per unit load such as articles packaged in containers. In the middle we have plotted the ‘forward pick storage only’ situation which is feasible for articles where the total required storage space fits in a forward pick slot. Both situations do not require time consuming replenishment operations. Most articles require a slot in the reserve storage and in the forward pick storage, as shown on the right, because the total required storage space is too large to fit in a forward pick slot. Preferably the articles with many handling requests for a small number of article units are assigned to forward pick slots if the total storage space in the forward pick storage is
3.1. The warehouse slot planning problem

scarce. A low number of handling requests generates only a few visits and
a high order frequency results in more replenishments.

In the case of single-address pick cycles, we can add the restriction to the
slot plan that each article is assigned to no more than one reserve location
and/or forward pick location because we gain nothing from more than
one assigned location when we can visit only one location during each
pick cycle.

3.1.2 Definition of the WSPP

With the handling models and the definitions of slot data and slot plan,
we can now formally define the warehouse slot planning problem.

**Definition 3.12** (Warehouse slot planning problem). Given is an
instance of slot data \( SD \in SD \), described in Definition 3.1. The problem is
to find a slot plan \( \chi : A \times T \rightarrow S \) and a utilization \( y : L \rightarrow \{0,1\} \) that
minimizes the cost function

\[
C_{WSPP} = w_h \cdot \sum_{a \in A} \sum_{t \in T} \Theta(a, t, \chi(a, t)) + w_u \cdot \sum_{l \in L} V(l) \cdot y(l) ,
\]

(3.19)

and satisfies the following two constraints.

I. The minimum required total and forward pick storage space of each
article in a planning period has to be met. Hence, for all \( a \in A \) and
\( t \in T \), we require

\[
\sum_{s \in \chi(a, t)} v(s) \geq w(a, t) , \quad \text{and} \quad (3.20)
\]

\[
\sum_{s \in \chi(a, t) \cap F} v(s) \geq w_f(a, t) . \quad (3.21)
\]

II. The total storage capacity of a location may not exceed a prescribed
maximum. Hence, for all \( l \in L \) and \( t \in T \), and

\[
K_{a,l,t} = \{ s \in \chi(a, t) \mid l_s(s) = l \} , \quad (3.22)
\]

the set of all slots in a location \( l \) that an article \( a \) is assigned to
during a planning period \( t \), we require

\[
\sum_{a \in A} \sum_{s \in K_{a,l,t}} v(s) \leq V(l) \cdot y(l) , \quad \text{and} \quad (3.23)
\]

\[
\sum_{a \in A} \sum_{s \in K_{a,l,t} \cap F} v(s) \leq V_f(l) . \quad (3.24)
\]

A slot plan \( \chi \) that satisfies (3.20)-(3.24) is called a *solution* for the slot
data \( SD \).

An instance of the WSPP is only feasible when for all planning periods
\( t \in T \) the minimum required total storage space is less than the total
available storage capacity, i.e., \( \sum_{a \in A} w(a, t) \leq \sum_{l \in L} V(l) \).
The first part of (3.19) evaluates the effect of the proposed slot plan on the total handling cost. We mentioned in Section 3.1.1, that the total handling time is an indicator for the total handling cost. Constraints on the total handling cost in a planning period or the maximum throughput time of a customer order in the warehouse are not included in this research. Ashayeri, Gelders & Van Wassenhove [1985] discuss models that include these handling constraints. The scheduling of the storage and retrieval requests with constraints on the release and due dates of the customer orders are taken into account at the decision level 'operations management'. Models that include these scheduling aspects are described by Lee & Kim [1995].

The cost contributed to the utilization of storage capacity are evaluated in the second part of (3.19). The storage capacity cost depend on the number of storage locations that are used during the year. The total storage capacity of the warehouse might be reduced when the locations remain unused during the whole year. Decisions on the maximum available storage capacity are taken at the decision level 'equipment and layout'. Instead of reducing the storage capacity, excess storage capacity can be used at that decision level to increase the service level and the flexibility of the warehouse. Temporary shortages in storage capacity should be avoided by renting additional storage capacity in the neighborhood of the distribution center. Warehouse sizing problems of this kind which use scenarios of minimal required storage space are discussed by Hung & Fisk [1984]. Rosenblatt & Roll [1988] studied the relation between service level and warehouse capacity with a simulation model.

3.1.3 Problem analysis of the WSPP

In this section, we analyse the WSPP in order to develop a solution approach that is capable of solving real world problem instances. After defining two subproblems of the WSPP, we prove that these subproblems of the WSPP are NP-hard. Then we may conclude that the WSPP itself is NP-hard too.

The main goal of the slot planning is to find a location and to allocate storage space at that location for each article during each planning period. We can split the WSPP into the following two subproblems, based on the two properties of a slot.

1. **Forward-reserve problem (FRP)**. Forward-reserve allocation tries to find the optimal storage space allocation between the forward pick
3.1. The warehouse slot planning problem

storage and the reserve storage with fixed, predetermined locations in each warehousing system. The space allocation is bounded by the total available storage capacity of the combined forward pick locations.

2. Stock location assignment problem (SLAP). Stock location assignment determines the optimal stock locations for each article in a planning period with fixed, predetermined storage space allocations in the reserve and forward pick storage.

The FRP is described by Hackman & Platzman [1990] and Hackman & Rosenblatt [1990], and the SLAP by Hausman, Schwarz & Graves [1976], both for automatic warehousing systems. We define both subproblems by reformulating the slot plan as an allocation plan or as an assignment plan.

3.1.4 The forward-reserve problem

For the forward-reserve problem we have to determine the reserve and forward pick storage space for each article. We reduce the WSPP by combining the locations of the forward pick slots into a single, combined location for the forward pick storage and the locations of the reserve slots into a single, combined location for the reserve storage. The combined location of the reserve storage includes also the forward pick slots, because we can use forward pick slots for reserve pick storage. The reserve storage capacity of a location \( l \in \mathcal{L} \) is therefore equal to the total storage capacity \( V(l) \) in the following definition of combined locations.

**Definition 3.13** (Combined locations). The set of aggregated locations \( \mathcal{W} = \{ x_r, x_f \} \subset \mathcal{L} \) consists of

- the location \( x_f \in \mathcal{L} \), which aggregates all locations \( l \in \mathcal{L} \) with forward pick storage capacity, i.e., \( V_f(l) > 0 \), has the aggregated forward pick storage capacity

\[
\hat{V}_f = \sum_{l \in \mathcal{L}} V_f(l), \text{ and} \tag{3.25}
\]

- the location \( x_r \in \mathcal{L} \), which aggregates all locations \( l \in \mathcal{L} \) with reserve storage capacity, i.e., \( V(l) > 0 \), has the aggregated total storage capacity

\[
\hat{V} = \sum_{l \in \mathcal{L}} V(l), \tag{3.26}
\]

where \( \mathcal{L} \) is the set of locations with total storage capacity \( V(l) \in \mathbb{N}^+ \) and forward pick storage capacity \( V_f(l) \in \mathbb{N}^+ \) for each \( l \in \mathcal{L} \). \qed
From the handling model of $\Theta$ in (3.1)-(3.18), it can be observed that the total handling time is independent of the amount of reserve storage space. Only the total time for replenishment $\Theta_{rpl}$ in the situation with reserve and forward pick storage, as formulated in (3.18), depends on the amount of forward pick storage space $V_5(\chi(a, t) \cap \mathcal{F})$. Hence, the allocated storage space in the reserve storage is the remainder of the minimum required total storage space after the allocation of storage space in the forward pick storage. This is expressed for all $a \in \mathcal{A}$ and $t \in \mathcal{T}$ by the equation

$$V_5(\chi(a, t) \cap \mathcal{R}) = \max(0, w(a, t) - V_5(\chi(a, t) \cap \mathcal{F})) . \quad (3.27)$$

When for each planning period the sum of the minimum required storage spaces is less or equal than the storage capacity of the forward pick storage, i.e., $\sum_{a \in \mathcal{A}} w(a, t) \leq \hat{V}_f$, then the allocation problem becomes trivial. In that case we can allocate for all articles the minimum required total storage space in the forward pick storage. Otherwise, we have to use storage capacity in both warehousing systems and the second part of (3.19) remains constant for all possible allocations. Therefore, we can remove the storage capacity utilization part and both weight factors from the cost function in the following definition of the forward-reserve problem.

**Definition 3.14** (Forward-reserve problem). An instance of the forward-reserve problem (FRP) is given by an eight-tuple $FD = (\mathcal{A}, \mathcal{T}, w, w_f, \mathcal{W}, \hat{V}, \hat{V}_f, \Theta)$, where $\mathcal{W} = \{x_r, x_f\}$ is the set of combined locations, $\hat{V} \in \mathbb{N}_0^+$ is the total storage capacity, and $\hat{V}_f \in \mathbb{N}_0^+$ is the storage capacity of the forward pick storage. The other sets and functions in $FD$ are described in Definition 3.1. We define an allocation plan as a pair $(v_r, v_f)$, where

$$v_r : \mathcal{A} \times \mathcal{T} \rightarrow \mathbb{N}_0^+ \quad (3.28)$$

allocates for an article in a planning period reserve storage space, and

$$v_f : \mathcal{A} \times \mathcal{T} \rightarrow \mathbb{N}_0^+ \quad (3.29)$$

allocates for each article in a planning period forward pick storage space. From these definitions we obtain the slot plan

$$\chi(a, t) = \{\{v_r(a, t), x_r\}, \{v_f(a, t), x_f\}\} ,$$

as described in Definition 3.2. The problem can now be formulated as to find an allocation plan $(v_r, v_f)$ that minimizes the cost function

$$C_{FRP} = \sum_{a \in \mathcal{A}} \sum_{t \in \mathcal{T}} \Theta(a, t, \chi(a, t)) , \quad (3.30)$$

and satisfies the following two constraints.
3.1. The warehouse slot planning problem

I. The minimum required total and forward pick storage space of an article during a planning period is met, i.e., for all \( a \in \mathcal{A} \) and \( t \in \mathcal{T} \),

\[
\begin{align*}
    v_r(a, t) + v_f(a, t) & \geq w(a, t) , \quad \text{and} \\
    v_f(a, t) & \geq w_f(a, t) .
\end{align*}
\]  

(3.31) (3.32)

II. The total storage capacity of a location may not exceed a prescribed maximum, i.e., for all \( t \in \mathcal{T} \),

\[
\begin{align*}
    \sum_{a \in \mathcal{A}} v_r(a, t) + v_f(a, t) & \leq \hat{V} , \quad \text{and} \\
    \sum_{a \in \mathcal{A}} v_f(a, t) & \leq \hat{V}_f .
\end{align*}
\]  

(3.33) (3.34)

An allocation plan \( (v_r, v_f) \) that satisfies (3.31)-(3.34) is called a solution for \( FD \).

It is easy to verify the existence of a feasible solution of the FRP in the case that all articles have only one possible storage space allocation. This can be shown by checking for each planning period \( t \in \mathcal{T} \) whether the sum of the sizes is less or equal to the aggregated forward pick storage capacity, i.e., \( \sum_{a \in \mathcal{A}} v_f(a, t) \leq \hat{V}_f \).

We reformulate the FRP as a knapsack problem in which we have to maximize the possible savings obtained by the allocation of forward pick storage space for each article. According to Martello & Toth [1991], there are a lot of solution techniques available for knapsack problems, which makes a knapsack formulation interesting. A similar approach is taken by Van den Berg [1996] for the FRP.

Remember that the allocated storage space in the forward pick storage is less or equal to the maximum unit load volume \( \omega_u \in \mathbb{N}^+ \). From this observation we can construct the following finite set of items.

**Definition 3.15** (Forward pick slots). The set of possible forward pick slots for an article \( a \in \mathcal{A} \) in a planning period \( t \in \mathcal{T} \) is equal to

\[
\mathcal{G}_{a, t} = \{ s \in \mathcal{F} \mid w_f(a, t) \leq v(s) \leq \omega_u \} ,
\]  

(3.35)

where \( \mathcal{F} \) is the set of forward pick slots with storage capacities \( v(s) \in \mathbb{N}_0^+ \) for each \( s \in \mathcal{F} \), \( w_f(a, t) \in \mathbb{N}_0^+ \) is the minimum required storage space of article \( a \) in planning period \( t \), and \( \omega_u \in \mathbb{N}^+ \) is the maximum unit load volume.

The set \( \mathcal{G} \) is the union of forward pick slots for all \( a \in \mathcal{A} \) and for all \( t \in \mathcal{T} \),
i.e.,

$$\mathcal{G} = \bigcup_{a \in A} \bigcup_{t \in \mathcal{T}} \mathcal{G}_{a,t},$$

(3.36)

where we assume that $\mathcal{G}_{i,t} \cap \mathcal{G}_{j,t} = \emptyset$ for all $i \neq j$. $\square$

Note that the forward pick slots in $\mathcal{G}$ are labeled for a specific article.

With three possible situations for allocating stock to reserve and forward pick storage we have to consider two combinations to calculate the savings compared with keeping all stock in reserve storage.

1. Moving all the stock from reserve storage to forward pick storage such that the minimum required total storage space is less or equal to the allocated forward pick storage space, i.e., $w(a, t) \leq v_f(a, t)$. This results in the so called 'forward pick storage only' situation.

2. A situation with replenishment results from moving a fraction of the stock from reserve to the forward pick storage such that the allocated forward pick storage space is larger than zero but still less than the minimum required total storage space, i.e., $0 < v_f(a, t) < w(a, t)$.

We can now formulate the savings in handling in the following definition.

**Definition 3.16** (Handling savings). The savings in handling $\Theta_{\text{save}}$ of assigning article $a \in A$ in planning period $t \in \mathcal{T}$ to forward pick slot $s \in \mathcal{G}_{a,t}$ is equal to

$$\Theta_{\text{save}}(a, t, s) = \begin{cases} 
\Theta_{\text{ro}} - \Theta_{\text{fo}} & \text{if } w(a, t) \leq v_f(a, t) \\
\Theta_{\text{ro}} - \Theta_{\text{rf}} & \text{if } 0 < v_f(a, t) < w(a, t) \\
0 & \text{otherwise},
\end{cases}$$

(3.37)

where $v_f(a, t) \in \mathbb{N}_0^+$ is the allocated forward pick storage space and $w(a, t) \in \mathbb{N}_0^+$ is the minimum required total storage space of an article $a$ in a planning period $t$. The handling time functions $\Theta_{\text{ro}}$, $\Theta_{\text{fo}}$, and $\Theta_{\text{rf}}$ are defined in (3.7)-(3.18). $\square$

The FRP for the forward pick storage with the use of savings leads to the following knapsack problem.

**Definition 3.17** (Knapsack forward-reserve problem). An instance of the knapsack forward-reserve problem (KFRP) is given by a six-tuple $KD = (A, \mathcal{T}, \hat{V}_f, \mathcal{G}, v, \Theta_{\text{save}})$, where $\hat{V}_f \in \mathbb{N}_0^+$ is the storage capacity of the forward pick storage, $\mathcal{G}$ is the set of possible forward pick slots for an article in a planning period, and $\Theta_{\text{save}}$ is a function that gives the savings
in handling time. The other sets and functions in KD are described in Definition 3.1. We define a knapsack assignment \( x_u \), where
\[
x_u : G \rightarrow \{0, 1\}
\]
assigns an article in a planning period to a forward pick slot. Here "1" corresponds to the situation that the specified slot is occupied in the forward pick storage and otherwise the specified slot is free, denoted by "0".

The problem is to find a knapsack assignment \( x_u \) that maximizes the total savings
\[
C_{KFRP} = \sum_{t \in T} \sum_{a \in A} \sum_{s \in G_{a,t}} \Theta_{save}(a, t, s) \cdot x_u(s)
\]
and that satisfies the following two constraints.

I. Exactly one quantity has to be allocated for each article \( a \in A \) and planning period \( t \in T \), hence
\[
\sum_{s \in G_{a,t}} x_u(s) = 1.
\]

II. The total storage capacity of the forward pick storage may not be exceeded, i.e., for all planning periods \( t \in T \)
\[
\sum_{a \in A} \sum_{s \in G_{a,t}} v(s) \cdot x_u(s) \leq \hat{V}_f.
\]

A knapsack assignment \( x_u \) that satisfies (3.40)-(3.41) is called a solution for KD.

The allocation of forward pick storage space can be carried out separately for each planning period because we made the allocation of storage space independent of the planning period. The problem for a single planning period resembles closely the multiple-choice knapsack problem (MCKP) described by Martello & Toth [1991]. The MCKP is also a 0-1 knapsack problem in which a partition of the total set of items is given, and it is required that exactly one item per subset is selected. In the KFRP, each subset \( G_{a,t} \) corresponds with the possible allocations for an article \( a \in A \) in a planning period \( t \in T \).

### 3.1.5 The stock location assignment problem

The goal of the stock location assignment problem (SLAP) is to find optimal stock locations for each article in the reserve and the forward pick storage. We require for this subproblem of the WSPP that the storage space allocations for the two warehousing systems are predetermined.
With predetermined storage space allocations, we can now formulate the SLAP as follows.

**Definition 3.18** (Stock location assignment problem). An instance of the stock location assignment problem (SLAP) is given by a 10-tuple \(LD = (\mathcal{A}, \mathcal{T}, v_r, v_f, \mathcal{L}, V, V_f, \Theta, w_h, w_u)\), where \(v_r\) is a function that gives the reserve storage space allocation and \(v_f\) is a function that gives the forward pick storage space allocation. The other sets and functions in \(LD\) are described in Definition 3.1. We define an assignment plan as a pair \((x_r, x_f)\), where

\[
x_r : \mathcal{A} \times \mathcal{T} \rightarrow \mathcal{L}
\]

(3.42)

assigns each article in a planning period to a reserve slot location, and

\[
x_f : \mathcal{A} \times \mathcal{T} \rightarrow \mathcal{L}
\]

(3.43)

assigns each article in a planning period to a forward pick slot location. From these definitions we obtain the slot plan

\[
\chi(a, t) = \{v_r(a, t), x_r(a, t)\}, \{v_f(a, t), x_f(a, t)\}
\]

as described in Definition 3.2. The problem is to find an assignment plan \((x_r, x_f)\) and a utilization \(y\) that minimizes the cost function

\[
C_{WSPP} = w_h \cdot \sum_{a \in \mathcal{A}} \sum_{t \in \mathcal{T}} \Theta(a, t, \chi(a, t)) + w_u \cdot \sum_{l \in \mathcal{L}} V(l) \cdot y(l),
\]

(3.44)

and satisfies the constraint that the total storage capacity of a location may not exceed a prescribed maximum, i.e., for all \(l \in \mathcal{L}\) and \(t \in \mathcal{T}\),

\[
\sum_{a \in \mathcal{A} | x_r(a, t) = l} v_r(a, t) + v_f(a, t) \leq V(l) \cdot y(l), \text{ and}
\]

\[
\sum_{a \in \mathcal{A} | x_f(a, t) = l} v_f(a, t) \leq V_f(l).
\]

(3.45)

(3.46)

An assignment plan \((x_r, x_f)\) that satisfies (3.45)-(3.46) is called a solution for \(LD\).

Note that the cost function of the SLAP in (3.44) is equal to the cost function of the WSPP, as given in (3.19).

We can solve the problem separately for the reserve and the forward pick slots because the replenishment time \(\bar{\theta}_{rpl}\) is independent of the relative location of the reserve and forward pick slots. Since forward pick storage may also be used for reserve pick storage as described by (3.45), we have to solve the SLAP for the forward pick storage before we can solve the SLAP for the reserve pick storage.
3.1.6 Complexity analysis of the WSPP

The complexity of the WSPP depends on the complexity of the subproblems KFRP and SLAP. When we prove that the KFRP and/or the SLAP are NP-hard, then we may conclude that also the WSPP is NP-hard.

Before analyzing the complexity of both subproblems, we introduce the following two well-known NP-complete problems, described among others by Garey & Johnson [1979].

**Definition 3.19 (Knapsack).** An instance of knapsack is given by a five-tuple $I = (\mathcal{U}, s, p, B, K)$, where $\mathcal{U}$ is a finite set of items with size $s(u) \in \mathbb{N}^+$ and profit $p(u) \in \mathbb{N}^+$ for each $u \in \mathcal{U}$, $B$ is a positive integer knapsack capacity and $K$ is a positive integer. The problem is to find a subset $\mathcal{U}' \subset \mathcal{U}$ such that $\sum_{u \in \mathcal{U}'} s(u) \leq B$ and such that $\sum_{u \in \mathcal{U}'} p(u) \geq K$. $\square$

**Definition 3.20 (Bin packing).** An instance of bin packing is given by a four-tuple $I = (\mathcal{U}, s, B, K)$, where $\mathcal{U}$ is a finite set of items with size $s(u) \in \mathbb{N}^+$ for each $u \in \mathcal{U}$, $B$ is a positive integer bin capacity and $K$ is a positive integer. The problem is to find a partition of $\mathcal{U}$ into disjoint sets $U_1, U_2, \ldots, U_K$ such that the sum of the sizes of the items in each $U_i$ is $B$ or less. $\square$

Karp [1972] proved that the knapsack problem is NP-complete and Garey & Johnson [1979] proved that the bin packing problem is NP-complete. We can prove the complexity of the decision versions of the KFRP and the SLAP straightforwardly by reduction to respectively knapsack and bin packing.

We examine the complexity of the the KFRP first. The decision version of the KFRP is defined as follows.

**Definition 3.21 (KFRPD).** An instance of the KFRPD is given by a seven-tuple $KD' = (\mathcal{A}, \tau, \hat{V}_f, \mathcal{G}, v, \Theta_{\text{save}}, K)$, where $\hat{V}_f \in \mathbb{N}_0^+$ is the storage capacity of the forward pick storage, $\mathcal{G}$ is the set of possible forward pick slots, $\Theta_{\text{save}}$ is a function that gives the savings in handling time, and $K \in \mathbb{N}^+$. The other sets and functions in $KD'$ are described in Definition 3.1. The problem is to find a knapsack assignment $x_u$, as described in Definition 3.17, such that

$$\sum_{t \in \tau} \sum_{a \in \mathcal{A}} \sum_{s \in G_{a,t}} \Theta_{\text{save}}(a, t, s) \cdot x_u(s) \geq K,$$  \hspace{1cm} (3.47)

and that satisfies (3.40)-(3.41). $\square$
Note that the determination of the allocation problem for a single article is trivial, since in that case the allocated space is equal to the minimum of the aggregated forward pick storage space and the maximum unit load volume or \( v_f(a, t) = \min(\hat{V}_f, \omega_u) \). We study the special case of the KFRPD with a single planning period and only two slot sizes for each article to prove that the KFRPD is NP-complete.

**Theorem 3.1.** The KFRPD is NP-complete.

**Proof.** The knapsack problem is a special case of the KFRPD. Let \( I = (\mathcal{U}, s, p, B, K) \) be an instance of the knapsack problem. We construct an instance \( I' = (\mathcal{A}, T, \hat{V}_f, \mathcal{G}, v, \Theta_{\text{save}}, K') \) of the KFRPD by choosing

- \( \mathcal{A} = \mathcal{U} \),
- \( T = \{t_1\} \),
- \( \hat{V}_f = B \),
- for all \( a \in \mathcal{A} \), we add an item \( g \) to \( \mathcal{G} \) with \( v(g) = s(a) \), and \( \Theta_{\text{save}}(a, t_1, g) = p(a) \) and an additional item \( g' \) to \( \mathcal{G} \) with \( v(g') = \Theta_{\text{save}}(a, t_1, g') = 0 \), such that \( \mathcal{G}_{a, t_1} = \{g, g'\} \),
- \( K' = K \).

The construction can be done in polynomial time. It can be verified that there exists a solution for \( I \) if and only if there exists a solution for \( I' \), which completes the proof. \( \square \)

The FRP has the same complexity as the knapsack version of the FRP, since the construction of an instance of the KFRP from an instance of the FRP and vice versa can be done in polynomial time.

Next we discuss the complexity of the SLAP. The decision version of the SLAP is defined as follows.

**Definition 3.22 (SLAPD).** An instance of the SLAPD is given by a seven-tuple \( LD' = (\mathcal{A}, T, v_r, v_f, \mathcal{L}, V, V_f) \), where \( v_r(a, t) \) is a function that gives the reserve storage space allocation and \( v_f(a, t) \) is a function that gives the forward pick storage space allocation of an article \( a \in \mathcal{A} \) in a planning period \( t \in T \). The function \( V(l) \) gives the total storage capacity and \( V_f(l) \) gives the forward pick storage capacity of a location \( l \in \mathcal{L} \), as described in Definition 3.1. The problem is to find an assignment plan \( (x_r, x_f) \), as described in Definition 3.18, that satisfies the constraint that the total storage capacity of a location may not exceed a prescribed
3.1. The warehouse slot planning problem

maximum, i.e., for all $l \in \mathcal{L}$ and $t \in \mathcal{T},$

$$\sum_{a \in A} x_r(a,t) = l \quad v_r(a,t) + v_f(a,t) \leq V(l), \text{ and}$$

$$\sum_{a \in A} x_f(a,t) = l \quad v_f(a,t) \leq V_f(l). \quad (3.48)$$

\[ \square \]

Instances of the SLAPD with a single location are trivial to solve. In the case of a single location, we can check in polynomial time if all articles fit into the location during all planning periods. We examine the case with a single planning period to prove that the SLAPD is NP-complete.

**Theorem 3.2.** The SLAPD is NP-complete.

**Proof.** The bin packing problem is a special case of the SLAPD. Let $I = (\mathcal{U}, s, B, K)$ be an instance of the bin packing problem. Construct an instance $I'' = (\mathcal{A}, \mathcal{T}, v_r, v_f, \mathcal{L}, V, V_f)$ of the SLAPD by choosing

- $\mathcal{A} = \mathcal{U},$
- $\mathcal{T} = \{t_1\},$
- for all $a \in \mathcal{A}, v_r(a,t_1) = s(a)$ and $v_f(a,t_1) = 0,$
- $|\mathcal{L}| = K,$
- $\mathcal{L} = \{l_1, l_2, \ldots, l_K\}$, and
- for all $l \in \mathcal{L}, V(l) = B$ and $V_f(l) = 0.$

The construction can be done in polynomial time. It can be verified that there exists a solution for $I$ if and only if there exists a solution for $I''$, which completes the proof. \[ \square \]

The SLAPD with one or more planning period is at least as hard to solve as the special case with a single planning period.

Due to the NP-completeness of FRP and SLAP, it is difficult to find an efficient solution approach for the WSPP. We remark that a typical distribution center for perishables, such as the distribution center of the wholesaler described in Section 5.2, has to deal with 5000 articles and at least as many locations, which results in very large instances of the WSPP. In order to solve these instances of the WSPP, we developed a decomposition strategy, based on both subproblems.

### 3.1.7 Decomposition strategies for the WSPP

In this section we introduce two decomposition strategies to handle the warehouse slot planning problem, based on two different aspects of the
solution quality, i.e., handling and space utilization. In a slot plan, we have to allocate storage space and find a location for the articles at the same time. In the previous section we showed that these decisions can be addressed separately by using the subproblems FRP and SLAP instead of the WSPP. The decomposition strategy determines the way we apply these subproblems in a solution approach for the WSPP. We have chosen the following two decomposition strategies.

1. *Handling first.* The handling oriented strategy tries to minimize the expected handling time by fixing the assigned location of each article throughout the year. The utilization cost is omitted from the cost function of the WSPP.

2. *Space first.* The space oriented strategy focuses on the utilization of the storage capacity. A slot plan that utilizes only a small part of the warehouse is called a compact slot plan. Such a compact slot plan has short travel distances which may reduce the total handling effort.

![Diagram](image)

Figure 3.3: The solution strategy for the warehouse slot planning problem.

The general solution strategy is illustrated in Figure 3.3. Based on the slot data of the WSPP, we first apply the FRP for each planning period
to obtain the reserve and forward pick storage space allocations for each article. With the slot data and these allocations, we solve the SLAP, which results in the assignment of the articles to locations and therefore in a slot plan.

Both decomposition strategies use the KFRP in the first step to allocate storage space to the warehousing systems. Remember that the KFRP can be solved separately for each planning period and that the solution of the KFRP is independent of the storage capacity utilization.

The 'handling first' strategy uses for the next step a special case of the SLAP with fixed assignments throughout the year and no utilization function. This special case is denoted with SLAP|H and is defined as follows.

**Definition 3.23** (SLAP|H). An instance of the stock location assignment problem for handling (SLAP|H) is given by an eight-tuple $HD = (A, T, v_r, v_f, \mathcal{L}, V, V_f, \Theta)$, where $v_r$ is a function that gives the reserve storage space allocation and $v_f$ is a function that gives the forward pick storage space allocation. The other sets and functions in $LD$ are described in Definition 3.1. We define an assignment plan as a pair $(\hat{x}_r, \hat{x}_f)$, where

$$\hat{x}_r : A \rightarrow \mathcal{L}$$  \hspace{1cm} (3.50)

assigns each article to a reserve slot location, and

$$\hat{x}_f : A \rightarrow \mathcal{L}$$  \hspace{1cm} (3.51)

assigns each article to a forward pick slot location. From these definitions we obtain the slot plan

$$\chi(a, t) = \{(v_r(a, t), \hat{x}_r(a)), \{v_f(a, t), \hat{x}_f(a)\}\},$$

as described in Definition 3.2. The problem is to find an assignment plan $(x_r, x_f)$ that minimizes the cost function

$$C_{\text{SLAP|H}} = \sum_{a \in A} \sum_{t \in T} \Theta(a, t, \chi(a, t)),$$  \hspace{1cm} (3.52)

and satisfies the constraint that the total storage capacity of a location may not exceed a prescribed maximum, i.e., for all $l \in \mathcal{L}$ and $t \in T$,

$$\sum_{a \in A | \hat{x}_r(a) = l} v_r(a, t) + v_f(a, t) \leq V(l), \quad \text{and}$$  \hspace{1cm} (3.53)

$$\sum_{a \in A | \hat{x}_f(a) = l} v_f(a, t) \leq V_f(l).$$  \hspace{1cm} (3.54)

An assignment plan $(\hat{x}_r, \hat{x}_f)$ that satisfies (3.53)-(3.54) is called a solution for $HD$. \hfill \Box
Limiting the assignment plan for an article from a single location in a planning period to a single location all the year round also reduces the number of decision variables and thus the complexity of the problem. The allocation of storage space given by the solution of the KFRP is not influenced by the locations of the articles in the reserve or forward pick storage.

The second or space oriented strategy tries, given the storage space allocations resulting from the KFRP, to minimize the number of utilized locations. This can be achieved by setting the handling weightfactor \( \omega_h \) to zero in problem instances of the SLAP, described in Definition 3.18. We denote the problems where the instance is modified in this way as SLAP|U. We remark that the SLAP|U resembles even closer the bin packing problem than the original SLAP. This is caused by the fact that the SLAP|U is more focused on packing the articles in as few locations/bins as possible than the SLAP.

### 3.2 The warehouse slot planning problem for perishables

The extension of the WSPP that models the handling of products with a keeping quality change during storage and handling, such as perishables, requires specific additional data and functions. In this section we define quality change data that make the warehouse slot planning problem specific for perishables. We use the static keeping quality to model the keeping quality loss in the slot plan. Adjacent locations in the warehouse with the same storage conditions are combined in zones. Normally, a zone coincides with a storage room.

**Definition 3.24** (Quality change data). Given is an instance of slot data \( SD \in SD \) as defined in Definition 3.1. An instance of the quality change data is a 10-tuple \( QC_{SD} = (Q, q_a, Z, z_l, \zeta, \kappa, \varepsilon_p, \Lambda, \Lambda_{max}, \omega_q) \), where

- \( Q \), a set of quality change groups \( q \),
- \( q_a : A \to Q \) gives the quality change group of an article,
- \( Z \), a set of zones \( z \),
- \( z_l : L \to Z \) gives the zone of a location,
- \( \zeta : Z \to \mathbb{N}^+ \) gives the total volume of a zone,
- \( \kappa : Z \to \mathbb{N}^+ \) gives the ventilation rate or the number of times per day that the total volume of a zone is completely refreshed through ventilation,
- $\epsilon_p : Q \times Z \rightarrow \mathbb{R}_0^+$ gives the ethylene production of a quality change group in a zone,
- $\Lambda : Q \times Z \times \mathbb{R}_0^+ \rightarrow \mathbb{R}^+$ gives the relative keeping quality loss of a quality change group in a zone with a specific ethylene concentration,
- $\Lambda_{\text{max}} : Q \rightarrow \mathbb{R}^+$ gives the maximum allowed relative keeping quality loss of a quality change group, and
- $\omega_q \in \mathbb{N}_0^+$, a weightfactor for keeping quality.

The set of all instances of the quality change data for $SD \in SD$ is denoted by $QC_{SD}$.

### 3.2.1 Quality change models

In this subsection, we describe generalized models for ethylene production and keeping quality that can be used for all quality change groups. The presented equations are derived from empirical data from the Sprenger Institute [1983]. These quality change equations are adjusted and extended based on work carried out at ATO-DLO by Van Doorn & Tijskens [1991], Tijskens & Polderdijk [1996], Polderdijk, Boerrigter & Tijskens [1995] and Woltering & Harkema [1987]. They also developed advanced models for a limited number of quality change groups, which are founded on a better theoretical basis, and are validated by experiments at the research facilities of ATO-DLO. The development and validation of these models are open research topics.

For the quality change models at the decision level 'slot planning', the following assumptions were made with respect to the keeping quality of the articles in the distribution center.

- **Stock level.** The average volume of an article that is stored overnight in a slot is equal to the allocated storage space of that slot. Therefore, the ethylene concentration in a zone is directly related to the allocated storage space of the articles in that zone.

- **Quality change group.** All articles are assigned to a quality change group, based on the quality change properties of the article. The factors initial keeping quality and storage time are the same for all articles in a quality change group.

- **Keeping quality loss.** The occupied slot of an article with the worst storage conditions for that article determines the keeping quality loss. The upper bound on the keeping quality loss is the same for all articles in a quality change group.
- **Storage conditions.** The temperature, the ethylene concentration, and the accessibility of a zone are the only storage conditions that influence the keeping quality loss of the articles in the distribution center. The effects of packaging, odors, and other atmospheric conditions are not included in the models. The effect of temperature on the keeping quality in the absence of all other effects is called the reference keeping quality.

- **Effective storage volume.** The effective storage volume or free space of a zone is the physical storage volume reduced by the volume of the stored articles and increased by the ventilation rate. We assume that the volume of an article unit contains no free space.

We discuss the effect of the storage conditions on keeping quality, starting with a definition of static keeping quality.

**Definition 3.25** (Static keeping quality). The static keeping quality \( k \) of quality change group \( q \in Q \) in zone \( z \in Z \) with a specific ethylene concentration \( \varepsilon(z) \) is modelled by the function

\[
k(q, z, \varepsilon(z)) = (1 - k_e(q, z, \varepsilon(z))) \cdot (1 - k_h(q, z)) \cdot k_{ref}(q, z),
\]

where

- \( \varepsilon : Z \rightarrow \mathbb{R}_0^+ \) gives the ethylene concentration in a zone,
- \( k_e : Q \times Z \times \mathbb{R}_0^+ \rightarrow [0, 1] \in \mathbb{R}_0^+ \) describes the effect of ethylene on a quality change group in a zone for a specific ethylene concentration,
- \( k_h : Q \times Z \rightarrow [0, 1] \in \mathbb{R}_0^+ \) describes the effect of handling on a quality change group in a zone, and
- \( k_{ref} : Q \times Z \rightarrow \mathbb{R}^+ \) gives the reference keeping quality of a quality change group in a zone.

The static keeping quality depends on the reference keeping quality \( k_{ref} \), changed by the effects of ethylene \( k_e \) and handling \( k_h \). This way of modeling the static keeping quality was proposed by Polderdijk, Boerrigter & Tijskens [1995] for controlled and modified atmosphere. Remember that we omitted the effects of controlled atmosphere storage and modified atmosphere packaging in the quality change models. We assume that the keeping quality at a specific temperature is always reduced by the effects of ethylene and handling. This is not the case for controlled atmosphere storage and modified atmosphere packaging that can also improve the reference keeping quality.
Figure 3.4: The reference keeping quality as a function of the temperature for the quality change groups 'lettuce' and 'banana'. Remark that 'lettuce' has no second quality change process.

**Temperature.** Kopec [1983] states that temperature is the most important effect with respect to the keeping quality of perishables. The reference keeping quality is influenced by two quality change processes: aging and chilling injury. The aging or first quality change process speeds up with higher storage temperatures, and the chilling injury or second quality change process gets more severe at lower storage temperatures. The maximum initial keeping quality is included in the quality change group functions.

**Definition 3.26** (Reference keeping quality). The reference keeping quality $k_{\text{ref}}$ for quality change group $q \in Q$ in a zone $z \in Z$ can be modelled as

$$ k_{\text{ref}}(q, z) = \begin{cases} \varphi_{ta}(q) \cdot e^{-\varphi_{ta}(q) \cdot \tau(z)} & \text{if } \tau(z) \geq \tau_{\text{ref}}(q) \\ \varphi_{ta}(q) - \varphi_{tb}(q) \cdot \tau(z) & \text{otherwise,} \end{cases} $$

(3.56)

where

- $\tau : Z \rightarrow [0, 44] \in \mathbb{R}_0^+$ gives the temperature in a zone,
- \( \tau_{\text{ref}} : Q \rightarrow \mathbb{R}_0^+ \) gives the threshold temperature of a quality change group at or above which the first quality change process takes place,

- \( \varrho_{\text{ta}} : Q \rightarrow \mathbb{R}^+ \) gives the maximum keeping quality when the first quality change process of a quality change group takes place,

- \( \varrho_{\text{tb}} : Q \rightarrow \mathbb{R}^+ \) gives the decrease in keeping quality when the first quality change process of a quality change group takes place,

- \( \varrho_{\text{la}} : Q \rightarrow \mathbb{R}^+ \) gives the maximum keeping quality when the second quality change process of a quality change group takes place, and

- \( \varrho_{\text{lb}} : Q \rightarrow \mathbb{R}^+ \) gives the decrease in keeping quality when the second quality change process of a quality change group takes place.

All temperatures in the model are expressed in \( ^\circ \text{C} \). Two examples of the dependence of the reference keeping quality with respect to the temperature are shown in Figure 3.4.

**Interaction.** In general, the interaction between various products is modelled by agents such as *odors* and *ethylene*. For example, the odor of onions is transferred by a different agent compared to the odor transfer of melons. Furthermore, some products produce the agent and some are susceptible to the agent. The levels of production and sensitivity of the agent are specific for the quality change group and dependent on the storage temperature. Moreover, the product that produces the agent may also be susceptible to it at the same time.

For simplicity, we restricted ourselves to the effects of the agent ethylene in the presented quality change models. As we already mentioned, the ethylene concentration in a zone depends on the slot plan and the ethylene production of the quality change groups that are assigned to the zone.

**Definition 3.27** (Ethylene production). The ethylene production \( \varepsilon_p \) of quality change group \( q \in Q \) in zone \( z \in Z \) with a temperature \( \tau(z) \) can be modelled by

\[
\varepsilon_p(q, z) = \begin{cases} 
\frac{\bar{\rho} \cdot \tau(z)}{20} 10^{\varrho_{\text{ep}}(q)-1} & \text{if } \varrho_{\text{ep}}(q) > 0 \\
0 & \text{otherwise,}
\end{cases} \quad (3.57)
\]

where

- \( \bar{\rho} \in \mathbb{R}^+ \), the average mass per unit of storage space, and

- \( \varrho_{\text{ep}} : Q \rightarrow [0, 4] \in \mathbb{N}_0^+ \) gives the ethylene production class of a quality change group.
3.2. *The WSPP for perishables*

With respect to the ethylene production class function \( \varrho_{\text{ep}} \), we distinguish five different classes of ethylene production. The class with no production corresponds with \( \varrho_{\text{ep}} = 0 \), and the class with the highest production with \( \varrho_{\text{ep}} = 4 \). For example, mushrooms produce no ethylene and bananas and tomatoes are examples of products with high ethylene production. In our model, the ethylene production is equal to zero at 0° C and increases linear with respect to the temperature of the zone.

The ethylene concentration in a zone depends on the number of quality change groups assigned to and present in the zone and the total volume of the zone. The volume of the zone has to be adjusted with respect to the volume of the quality change groups that are assigned to it and that are present in the zone, and with respect to the rate of ventilation. A ventilation rate of 2 means that the effective volume of the zone also increases by a factor of 2.

**Definition 3.28** (Ethylene concentration). The ethylene concentration \( \varepsilon \) in zone \( z \in Z \) is given by

\[
\varepsilon(z) = \frac{\sum_{q \in Q} \varepsilon_{\text{p}}(q, z) \cdot \tilde{w}_q(q, z)}{\kappa(z) \left( \zeta(z) - \sum_{q \in Q} \tilde{w}_q(q, z) \right)},
\]

where \( \tilde{w}_q : Q \times Z \to \mathbb{N}_0^+ \) represents the volume of a quality change group that is stored overnight in a zone, \( \varepsilon_{\text{p}} \) gives the ethylene production of a quality change group in a zone, and \( \kappa(z) \) is the ventilation rate and \( \zeta(z) \) the volume of zone \( z \).

High utilization and low ventilation of a zone may result in a high ethylene concentration, even when the ethylene production is relatively low. When the average stock level in the slots is only 50 % of the allocated storage space, we get the same ethylene concentration according to (3.58) by reducing the physical volume of the zone \( \zeta \) with 50 %.

**Definition 3.29** (Ethylene effect). The ethylene effect on the keeping quality \( k_e \) of a quality change group \( q \in Q \) in zone \( z \in Z \) with a temperature \( \tau(z) \) and a ethylene concentration \( \varepsilon(z) \) is given by

\[
k_e(q, z, \varepsilon(z)) = \frac{\max(0, \min(1, \varrho_{\text{ea}}(q) \cdot \ln(\varepsilon(z)) - \varrho_{\text{eb}}(q)))}{1 + e^{-0.8(\tau(z) - 5)}} ,
\]

where \( \varrho_{\text{ea}}, \varrho_{\text{eb}} : Q \to \mathbb{R}_0^+ \) give the ethylene sensitivity of a quality change group.
The ethylene effect on the keeping quality behaves like a S-curve in relation to the ethylene concentration and the storage temperature. The logarithmic expression in the numerator of (3.59) explains why the ethylene sensitivity changes from 0 to 1 within a small range of the ethylene concentration in the zone. Typical values lie between 1.5 and 3.5 ppm. The expression with the storage temperature in the denominator of (3.59) makes that the ethylene effect changes dramatically around 5° C. Models for other types of agents resemble closely this ethylene model.

The effect of ethylene on the static keeping quality for two different quality change groups is shown in Figure 3.5.

Handling. The effect of handling on the keeping quality depends primarily on the quality change group. Handling causes shock and vibration and thus stress to fresh produce. Strawberries are an example of products that are highly susceptible with respect to handling. Processed foodstuffs are seldom susceptible to handling.

**Definition 3.30** (Handling effect). The handling effect \( k_h \) on a quality change group \( q \in Q \) in zone \( z \in Z \) is modelled by

\[
k_h(q, z) = \frac{\varphi_h(q)}{1 + \sigma(z)},
\]

(3.60)

where

- \( \varphi_h : Q \to [0, 1] \in \mathbb{R}_0^+ \) gives the maximum handling sensitivity of a quality change group, and

- \( \sigma : Z \to [0, 1] \in \mathbb{R}_0^+ \) gives the accessibility of a zone. Here "0" corresponds to an inaccessible zone and "1" to a perfectly accessible zone.

In (3.60), the effect of the handling sensitivity of a quality change group on the keeping quality increases when the zone is less accessible. In a perfectly accessible zone, the effect remains 50% of the maximum handling sensitivity \( \varphi_h \). Remark that we used in (3.60) the accessibility of a zone as a measure for the amount of handling. Hence, low accessibility results in additional handling compared with high accessibility. The models for vibration and shock developed by Yang & Prussia [1992] are more suitable for transportation than for storage.

**Relative keeping quality.** The keeping quality loss in the warehouse depends on the initial keeping quality when the product enters the ware-
3.2. The WSPP for perishables

![Graphs showing static keeping quality as a function of temperature and various ethylene concentrations for lettuce and banana.]  

Figure 3.5: The static keeping quality as a function of temperature and various ethylene concentrations for the quality change groups ‘lettuce’ and ‘banana’. The quality change group ‘banana’ has two local optima when the ethylene concentration rises above 2 ppm.
house. The initial keeping quality just after the harvest or production changes in the distribution chain with the storage conditions and the duration of stay in the different stages in the distribution chain. Tijskens & Polderdijk [1996] discuss these dynamic aspects of quality change models for vegetables and fruits and their importance for the simulation of the distribution chain of perishables. With simulation models of the distribution chain that incorporate dynamic quality change models, we can predict the effect of changes in the distribution on the keeping quality of the products. Broekmeulen [1996b] proposes a method of incorporating dynamic quality change models in distribution network models that uses routes from production to consumption with completely specified storage conditions instead of partial routes or arcs between the different stages or nodes in the distribution network.

Since the warehouse acts as a decoupling point in the distribution of perishables, we are not able to know in sufficient detail the history and the future of the products that pass the warehouse. Therefore, we need a measure of keeping quality loss in a warehouse that is independent of the initial keeping quality of the products, the duration of stay in the warehouse, and the remaining time of the products in the distribution chain until consumption by the end consumers. We propose a measure that relates the keeping quality loss in a specific slotplan to the keeping quality loss at ideal conditions. In (3.55), the ethylene concentration in a zone depends on the slot plan. The relative keeping quality loss is the decrease in keeping quality compared with the keeping quality at the reference temperature \( \tau_{\text{ref}} \) without any effects of handling or ethylene.

**Definition 3.31** (Relative keeping quality). The relative keeping quality loss \( \Lambda \) of quality change group \( q \in Q \) in zone \( z \in Z \) with a specific ethylene concentration is given by

\[
\Lambda(q, z, \varepsilon(z)) = 1 - \frac{k(q, z, \varepsilon(z))}{k_{\text{ref}}(q, z')} ,
\]

(3.61)

where \( k \) gives the static keeping quality loss, \( k_{\text{ref}} \) the reference keeping quality, and the temperature \( \tau(z') \) in the reference zone \( z' \in Z \) is equal to the threshold temperature \( \tau_{\text{ref}}(q) \) of the quality change group \( q \).

In the proposed model, the relative keeping quality loss is equal to the fraction of the initial keeping quality that is lost every day compared to the storage at the ideal conditions for that quality change group. A relative keeping quality loss of 50% in a warehouse implies that during
3.2. The WSPP for perishables

each day the product is stored in the distribution center, the shelf life is reduced with one and a half day instead of just one day. In this way, the subsequent stages in the distribution chain have less time to deliver the product to the consumer. When the reference keeping quality corresponds with a shelf life of less than one day, the storage of that quality change group is impractical. This is caused by the observation that for this small keeping quality the total shelf life gets lost during storage, even at ideal conditions.

3.2.2 Definition of the WSPP for perishables

The warehouse slot planning problem for perishables (WS3P) is formulated by extending the WSPP by both an cost function that handles the effects on keeping quality and by constraints that describe the maximum allowed keeping quality loss. For the WS3P, we made the following assumptions.

- **Effect of keeping quality loss.** The effect of the keeping quality loss of an article on the slot plan is determined by the assignment of that article to the slot with the worst storage conditions for that article. Since at the tactical level we do not know exactly which of the assigned slots will be used during the daily operations, we have to consider the worst scenario.

- **Article importance.** The keeping quality loss of all articles in the assortment is equally important. When this is not the case, we can introduce a weight factor for each individual article based on factors that distinguish the articles such as added value and yearly turnover.

Before we can define the WS3P, we need to determine the zone where a slot can be found since a slot receives the storage conditions of a zone in slot planning.

**Definition 3.32 (Zone of slot).** The zone of a slot is given by \( z_s(s) = z_l(l_s(s)) \) where \( z_s : S \rightarrow Z \) gives the zone of a slot \( s \in S \) and \( l_s(s) \) gives the location \( l \in L \), as defined in Definition 3.1. The function \( z_l(l) \) gives the zone of a location \( l \).

**Definition 3.33 (WSPP for Perishables).** An instance of the warehouse slot planning problem for perishables (WS3P) is given by a pair \((SD, QC_{SD})\), with slot data \( SD \in \mathcal{SD} \) as described in Definition 3.1 and quality change data \( QC_{SD} \in QC_{SD} \) as described in Definition 3.24. The
problem is to find a slot plan $\chi$ and a utilization $y$, as defined in Definition 3.2, that minimize the cost function

$$C_{WS3P} = \omega_h \cdot \sum_{a \in \mathcal{A}} \sum_{t \in \mathcal{T}} \Theta(a, t, \chi(a, t)) + \omega_u \cdot \sum_{l \in \mathcal{L}} V(l) \cdot y(l)$$

$$+ \quad \omega_q \cdot \sum_{a \in \mathcal{A}} \max_{t \in \mathcal{T}} \left( \max_{s \in \chi(a, t)} \Lambda(q_a(a), z_a(s), \varepsilon(z_a(s), t)) \right), \quad (3.62)$$

and that satisfies (3.20)-(3.24). Furthermore, the following constraints have to be satisfied too.

The keeping quality loss of an article in an assigned slot should be less than the maximum allowed relative keeping quality loss of the considered quality change group. Hence, this requirement can be formulated as

$$\Lambda(q_a(a), z, \varepsilon(z, t)) \leq \Lambda_{\max_a}(q_a(a)) \quad \text{if} \quad \mathcal{Y}_{a,z,t} \neq \emptyset, \quad (3.63)$$

for all $a \in \mathcal{A}$, $z \in \mathcal{Z}$ and $t \in \mathcal{T}$. Here,

$$\mathcal{Y}_{a,z,t} = \{s \in \chi(a, t) \mid z_a(s) = z\} \quad (3.64)$$

is the set of slots in zone $z$ to which an article $a$ can be assigned during planning period $t$.

Remember that the keeping quality loss depends on the ethylene concentration in a zone.

$$\varepsilon(z, t) = \frac{\sum_{a \in \mathcal{A}} \sum_{s \in \mathcal{Y}_{a,z,t}} \varepsilon_p(q_a(a), z) \cdot v(s)}{\kappa(z) \left( \zeta(z) - \sum_{a \in \mathcal{A}} \sum_{s \in \mathcal{Y}_{a,z,t}} v(s) \right)} \quad (3.65)$$

is a function that models the ethylene concentration in a zone $z$ in a planning period $t$. This function results from (3.58) and the assumption that the allocated storage space is equal to the average amount of quality change group present in the zone.

A slot plan $\chi$ that satisfies (3.20)-(3.24) and (3.63)-(3.65) is called a solution for $(SD, QC_{SD})$. \hfill \Box

The third part of (3.62) evaluates the effect of the proposed slot plan on the keeping quality loss. Note that the value of the maximum allowed relative keeping quality loss depends on the initial keeping quality, the number of days in stock at the distribution center and the minimum required keeping quality at the subsequent stages in the distribution chain. We assumed that these factors are the same for each article in a quality change group. The constraint on the maximum allowed relative keeping quality loss may
be used to guarantee that the warehouse does not waste an excessive amount of the initial keeping quality during storage. This constraint limits the trade-off in keeping quality between the different quality change groups in the slot plan. In this way a reduction in keeping quality loss of one quality change group may only be exchanged with a larger reduction in keeping quality loss of another quality change group when the constraint on the maximum allowed relative keeping quality loss is not violated.

3.2.3 Problem analysis of the WS3P

In this section, we analyse the WS3P for the same reason as we have analyzed the WSPP i.e., the development of a solution approach that is capable of solving real world problem instances. After defining a special case of the WS3P, we prove that this special case of the WS3P is NP-hard. From this we conclude that the WS3P itself is NP-hard too.

For the solution approach of the WS3P, we propose a special case of the WS3P that reduces the problem size. From the previous subsection it follows that the main extension of the WS3P compared to the WSPP is the assignment of quality change groups to zones. Hence, we can construct a special case of the WS3P, called the assignment problem for perishables (APP), by aggregating articles to quality change groups and locations to zones. Moreover, each quality change group should be assigned to just one zone. The aggregation of both articles in quality change groups and locations in zones reduces the problem size of the WS3P. We remark that a typical distribution center for perishables, such as the distribution center of the wholesaler described in Section 5.2, has to deal with 5000 articles and at least as many locations, which can be reduced to 200 quality change groups and not more than 10 zones.

3.2.4 The assignment problem for perishables

In order to construct the APP as a special case of the WS3P, we first combine the storage capacities of the slots in a zone to a so called zone capacity.

Definition 3.34 (Zone capacity). The zone capacity $V_z$ of zone $z \in Z$ is the combined storage capacity $V(l)$ of the locations $l \in L$ in that zone, i.e.,

$$V_z(z) = \sum_{l \in L, z_l(l) = z} V(l).$$
Secondly, we have to combine the total storage space requirement of articles in a storage space requirement for the corresponding quality change group.

**Definition 3.35** (Group storage space requirement). The group storage space requirement \( w_q \) of quality change group \( q \in \mathcal{Q} \) in planning period \( t \in \mathcal{T} \) is the combined minimum required total storage space \( w(a, t) \) of the articles \( a \in \mathcal{A} \) that belong to that quality change group, i.e.,

\[
w_q(q, t) = \sum_{a \in \mathcal{A} | q_a(a) = q} w(a, t) .
\]

We define the assignment problem for perishables by reformulating the slot plan as an assignment plan for the quality change groups. For this aggregated problem, we omit the handling cost and the utilization cost.

**Definition 3.36** (Assignment problem for perishables). An instance of the assignment problem for perishables (APP) is given by a 11-tuple \( AD = (\mathcal{Q}, \mathcal{T}, w_q, \mathcal{Z}, V_z, \zeta, \kappa, \varepsilon_p, \Lambda, \Lambda_{\max}, \omega_q) \), where \( w_q(q) \in \mathbb{N}_0^+ \) gives the group storage space requirement for each quality change group \( q \in \mathcal{Q} \), and \( V_z(z) \in \mathbb{N}_0^+ \) gives the zone capacity of each zone \( z \in \mathcal{Z} \). The other sets and functions are described in Definition 3.33. We define the assignment for perishables as

\[
x_z : \mathcal{Q} \times \mathcal{T} \rightarrow \mathcal{Z} ,
\]

i.e., a function that assigns a quality change group to a zone for each planning period. The problem is to find an assignment for perishables \( x_z \) that minimizes the cost function

\[
C_{\text{APP}} = \omega_q \sum_{q \in \mathcal{Q}} \max_{t \in \mathcal{T}} \Lambda(q, x_z(q, t), \varepsilon(x_z(q, t), t)) ,
\]

and satisfies the following two constraints.

I. The storage capacity of the zone may not be exceeded, i.e., for all \( z \in \mathcal{Z} \) and \( t \in \mathcal{T} \),

\[
\sum_{q \in \mathcal{Q} | x_z(q, t) = z} w_q(q, t) \leq V_z(z) .
\]

II. The keeping quality of a quality change group in an assigned zone should always be lower than the maximum allowed relative keeping quality loss of the quality change group when the group storage
space requirement of the quality change group is greater than zero, i.e., for all \( q \in Q \) and \( t \in T \) with \( w_q(q,t) > 0 \), we require

\[
\Lambda(q, x_z(q,t), \varepsilon(x_z(q,t), t)) \leq \Lambda_{\text{max}}(q). \tag{3.69}
\]

Here,

\[
\varepsilon(z, t) = \frac{\sum_{q \in Q|x_z(q,t)=z} \varepsilon_p(q,z) \cdot w_q(q,t)}{\kappa(z) \left( \zeta(z) - \sum_{q \in Q|x_z(q,t)=z} w_q(q,t) \right)} \tag{3.70}
\]

describes a function that models the ethylene concentration in a zone \( z \) in a planning period \( t \).

An assignment for perishables \( x_z \) that satisfies (3.68)-(3.70) is called a solution for \( AD \).

Like the SLAP, the APP is in essence a generalized assignment problem. The complexity analysis of the APP is analogue to the discussion of the complexity of the SLAP. The decision version of the APP is defined as follows.

**Definition 3.37 (APPD).** An instance of the APPD is given by a five-tuple \( AD' = (Q, T, w, Z, V_z) \), where \( w_q(q) \in \mathbb{N}_0^+ \) gives the group storage space requirement for each quality change group \( q \in Q \), and \( V_z(z) \in \mathbb{N}_0^+ \) gives the zone capacity of each zone \( z \in Z \). The other sets and functions are described in Definition 3.33. The problem is to find an assignment for perishables \( x_z \), as described in Definition 3.33, that satisfies (3.68).

Note that the APPD only contains integers. In the case, when there is a single quality change group or a single zone, the solution is trivial. In these cases, the problem can be solved straightforward because of the absence of alternatives. For the other cases, even when we have a single planning period, we can prove straightforwardly by reduction from bin packing that the decision version of the APP is NP-complete.

**Theorem 3.3.** The APPD is NP-complete.

*Proof.* Bin packing problem reduces in polynomial time to the APPD with a single planning period. This can be seen as follows. Let \( I = (U, s, B, K) \) be an instance of the bin packing problem. Construct an instance \( I'' = (Q, T, w, Z, V_z) \) of the APPD by choosing

- \( Q = U \),
- \( T = \{t_1\} \),
for all $q \in Q, w_q(q, t_1) = s(q),$

- $|Z| = K,$

- $Z = \{z_1, z_2, \ldots, z_K\},$ and

- for all $z \in Z, V_z(z) = B.$

The construction can be done in polynomial time. It can be verified that there exists a solution for $I$ if and only if there exists a solution for $I''$, which completes the proof. \hfill \Box

The APPD with one or more planning periods is at least as hard to solve as the special case with a single planning period. The dependence of the quality change models on the ethylene concentration in the APP make the problem even harder to solve. The NP-completeness makes the APP very resistant to efficient solution approaches.

3.2.5 Decomposition strategy for the WS3P

In this section we extend the decomposition strategies of the WSPP to meet the special requirements of the WS3P. In a slot plan for the WS3P, we need to determine the storage conditions for the slot besides allocating storage space and finding a location.

The generalized solution strategy for the WS3P starts with an assignment procedure, that assigns each quality change group to the zone where the keeping quality loss is minimal, as is illustrated in Figure 3.6. For each zone, we apply the FRP to obtain the reserve and forward pick storage space allocations and subsequently the SLAP to assign the articles to locations. In this way, the solution strategy results in an assignment of articles to slots and therefore a slot plan for the WS3P. Remark that this strategy is an extension of the solution strategy of the WSPP.

In Section 3.1.7, we introduced the decomposition strategies ‘handling first’ and ‘space first’. The handling oriented strategy requires fixed assignments of the articles throughout the year. Therefore we also need fixed assignments of the quality change groups in the APP. This is achieved by replacing in Definition 3.36 the function $x_2(q, t)$ with the function

$$\hat{x}_2 : Q \rightarrow Z .$$ (3.71)

The modified problem is denoted by APP|F. The original version can be used unchanged for the space oriented strategy and is renamed to APP|V.
3.2. The WSPP for perishables

![Diagram](image)

Figure 3.6: The solution strategy for the warehouse slot planning problem for perishables.

### 3.2.6 Extensions of the APP

One of the aims of the WS3P is to reduce the keeping quality loss by assigning quality change groups to zones with predefined zone capacities and predefined storage conditions. Both the zone capacities and the storage conditions in the zones have an effect on the keeping quality loss and therefore on the assignment plan. We propose extensions of the WS3P that change the predefined zone temperature and/or predefined zone capacity in decision quantities.

We use the APP in the solution strategy of the WS3P to assign articles to zones with predefined storage capacity and storage conditions. With the APP|V the articles can get fluctuating assignments with respect to storage space requirements during the year, caused by seasonal effects. This makes it less interesting to adapt the zone capacity and the storage conditions.

In the case of the APP|F, which results in fixed assignments throughout the year, we consider the following two types of problem extensions.
- **Storage condition adjustment.** When the storage temperature is changed to a temperature that is more suited with respect to all or a subset of the articles during a planning period, the keeping quality loss is reduced without the need for additional storage capacity or handling. Remark that more ventilation in a zone increases the energy consumption, but reduces the interactions between the articles.

- **Layout redesign.** Changing the storage capacity of the zones in the original layout might improve the match between available and required storage capacity. The storage conditions of a zone, for example the temperature, might be adjusted at the same time. The associated volume change of a zone has also an effect on the ethylene concentration.

In the case of storage condition adjustment, we focus only on the effect of the temperature since this zone condition is the easiest to adjust in existing storage rooms. For example, changes in the ventilation requires particular adaptations to the building and hence additional investments. The other zone conditions such as relative humidity are much harder to adjust. For both the above described types of problem extensions, we can use the function

$$\tau_z : \mathcal{Z} \rightarrow \mathbb{R}_0^+$$

as decision variable in the definition of the APP|F. This new function replaces the temperature function $\tau$ that is used in the quality change models to calculate the relative keeping quality loss $\Lambda$.

The redesign of the layout, as proposed by the second type of problem extension, is determined indirectly by adding the function

$$V'_z : \mathcal{Z} \rightarrow \mathbb{N}^+,$$

i.e., a function that gives the allocated storage capacity to each zone. This function replaces the zone capacity function $V_z$. To ensure that redesigned layouts stay within the limits of the total available storage capacity of the warehouse $\hat{V} \in \mathbb{N}^+$, we have to add the constraint

$$\sum_{z \in \mathcal{Z}} V'_z(z) = \hat{V}$$

(3.74)

to the problem definition of APP|F. We determine the variable zone volume $\zeta_z(z)$ from the initial zone capacity $V_z(z) \in \mathbb{N}^+$ and zone volume $\zeta(z) \in \mathbb{N}^+$ for each $z \in \mathcal{Z}$ as follows

$$\zeta_z(z) = \frac{\zeta(z)}{V_z(z)} \cdot V'_z(z).$$

(3.75)
3.3. Overview of the hierarchical decomposition strategy

Hereby, we assumed that the zone volume is always a fixed multiple from the zone capacity.

Using the zone capacity function $V'_z$ in the APP|F gives the opportunity to investigate alternative layouts of the warehouse. With the temperature function $\tau_2$ in the APP|F we can investigate different storage conditions.

![Diagram](image)

Figure 3.7: The overall hierarchical decomposition of the layout and operations management problem for a distribution center for perishables, with special attention for the slot planning decision level.

### 3.3 Overview of the hierarchical decomposition strategy

The problems defined for the decision level ‘slot planning’ have several links with the decision levels ‘equipment and layout’ and ‘operations management’. The storage conditions and the number of zones chosen in the layout influence the slot plan. The choice of equipment has to meet the storage space requirement of the slot plan. The workload that a slot plan generates determines the amount of personnel that is needed to meet the agreed lead time of the customer orders. Short throughput times can be achieved with efficient and effective operating policies and/or a large number of pickers. The efficiency of the operating policies determine on the
other hand the possible types of slot plans, such as fixed slots or variable assignments throughout the year.

The general solution strategy for slot planning with its relations with the decision levels ‘equipment and layout’ and ‘operations management’ is summarized in Figure 3.7. We propose to solve the formulated NP-hard subproblems with local search techniques, which we introduce in Chapter 4. The assignment of perishables to zones is investigated in Chapter 5. In Chapter 6 we study the forward-reserve allocation with the KFRP. The differences between the handling and the space oriented strategy for the stock location assignment is discussed in Chapter 7.
Local search

In this chapter, we present local search as a family of approximative solution techniques or algorithms for the NP-hard problems that we have formulated in the previous chapter for the decision level 'slot planning'. We introduce some popular local search techniques, using a general framework which is based on the research of Vaessens, Aarts & Lenstra [1996] and Aarts & Lenstra [1997]. We discuss the development and implementation of these techniques in our research and the methods that we applied to handle the constraints in our problems.

4.1 Introduction

Local search is a solution process that tries to improve a given initial solution by making relatively small changes in several steps in the solution space. The quality of the solutions is determined with the object or cost function of the problem. As discussed by Bounds [1987] and Glover & Greenberg [1989], several local search techniques find their origin in nature. We distinguish the following four elements in each local search algorithm: initial solution construction, neighborhood function, acceptation criterion and stop criterion. These four basic elements and their interrelations are illustrated in Figure 4.1.
During the initial solution construction an initial solution is generated by assigning random values to the decision quantities or by a specific construction algorithm that constructs a feasible solution based on a greedy or heuristic rule.

During the iterative improvement the following cycle is carried out several times. At the first step, a neighborhood function generates candidate solutions or neighbors from the current solution. A neighborhood graph connects neighboring solutions. A neighborhood graph is strongly connected when each solution can be reached from any other one. A neighborhood function consists of moves that create a candidate solution with a small change in the current solution. One has to find a trade-off between a small neighborhood that can be searched quickly, and a large neighborhood with a higher probability of good neighbors but larger search times. The choice of a neighborhood function depends on the type of local search algorithm and the specific problem structure.

In the second step of the iterative improvement, an acceptance criterion
4.1. Introduction

selects the new current solution for the next cycle. A pivot rule determines which candidate solution is selected from all neighboring solutions. Possible pivot rules are first improvement and best improvement. First improvement stops the neighborhood search after the first neighbor that is better than the current solution. Best improvement takes the best neighbor after a search of the entire neighborhood. We remark that the acceptation criterion is the main difference between various local search algorithms.

A stop criterion determines when the iterative improvement is terminated. Possible stop criteria are a limited number of iterations, a limited amount of computation time or the absence of newly accepted solutions during the iterative improvement cycle. A local optimum has been reached if the object value of the last accepted solution is better or equal than all its neighbors. The number and the quality of the local optima depends on the combinatorial structure of the problem at hand.

Based on the general framework, we explain the differences and similarities between several commonly applied local search techniques, subsequently tabu search, threshold algorithms, and genetic algorithms.

**Tabu search.** In tabu search we restrict the set of candidate solutions by a tabu list. A tabu list defines implicitly or explicitly forbidden moves from the current solution. The tabu list is updated after each iteration. The length of the tabu list can vary during the search process. During intensification of the search, a smaller part of the search space is more intensely examined by making the tabu list shorter. New, perhaps unexplored regions of the search space are searched during diversification. We diversify the search by making the tabu list longer or by performing several random restarts. An aspiration criterion can overrule the tabu status of a move. More details about tabu search can be found in the tutorial by Glover [1990].

**Threshold algorithms.** We accept a new solution in threshold algorithms when the difference in object value between the current and the candidate solution is below a certain threshold. Dueck & Scheuer [1990] describe such an algorithm. When the threshold value is equal to 0, only true improvements are accepted which is the case in classical iterative improvement. We denote iterative improvement with multi-start capabilities as repeated descent. Multi-start means that if the search gets stuck in a local optima before the stop criterion is met then we perform a random restart and the search continues. Nonnegative thresholds which gradually
decrease during the search process to 0 are used in *threshold accepting* algorithms. *Simulated annealing* as described by Aarts & Van Laarhoven [1985] uses nonnegative and stochastic thresholds that depend on a control parameter called $T$ or *temperature* and a stochastic variable $u \in (0,1] \subset \mathbb{R}$ in the following way: $-T \ln u$. The control parameter $T$ gradually decreases during the search process according to a *cooling schedule*. The cooling schedule determines the number of cycles between two adjustments of the control parameter. For each acceptance test of a candidate solution, $u$ is drawn anew from a uniform distribution on $(0,1]$. Simulated annealing technique has been introduced by Kirkpatrick, Gelatt & Vecchi [1983] and Černý [1985]. More details can be found in the reviews by Van Laarhoven & Aarts [1987] and Aarts & Korst [1989].

**Genetic algorithms.** Instead of a single current solution, genetic algorithms work with a set or *population* of current solutions at each iteration. The operator *mutation* executes a move using only one solution from the current population. The operator *recombination* or *crossover* uses the information of two or more solutions in the current population to construct candidate solutions. The operator *selection* acts as acceptance criterion in the combined set of current and candidate solutions and ensures that the population size remains constant during the search process. The idea for genetic algorithms comes from Holland [1975]. An excellent introduction to genetic algorithms is written by Goldberg [1989]. A recent review from Mattfeld [1996] explains most important aspects for applying genetic algorithms to scheduling problems. In *parallel descent*, the recombination operator is replaced by the *replacement* operator that uses only one solution from the current population. With replacement the values of a random number of decision quantities are replaced with randomly chosen new values within the range of the decision quantities.

Parallel descent and genetic algorithms are in potention parallel search techniques because we can distribute the population and the calculations over parallel processors. Sequential local search techniques such as repeated descent, simulated annealing, and tabu search operate on one current solution at a certain time.

The above mentioned local search algorithms can be combined to hybrid algorithms that exploit perhaps better the specific problem structure.
4.2 Implementation

For our computational studies of the slot planning problem, we implemented five different local search techniques: repeated descent, simulated annealing, tabu search, parallel descent and a genetic algorithm.

The chosen implementation of a local search technique, and settings of the parameters have been subject to many experimental configurations and trail runs for the different optimization models. In general we used the implementation described below, derived from the research literature on the NP-hard quadratic assignment problem (QAP). The NP-hard QAP is reviewed by Burkard [1984]. In some cases we deviated from this generic setup to incorporate specific properties of the slot planning problems. Comparable computational studies for combinatorial optimization problems are abundant in the literature. A good example is the research on the single machine scheduling problem by Crauwels, Potts & Van Wassenhove [1996].

For the majority of the considered problems, we used blind search as defined by Goldberg [1989]. This means that we do not use auxiliary information from the problem to select a move and that the complete object function is recalculated after each move. Partial recalculation of the object function was either difficult to implement or even impossible because of the problem structure.

The initial solutions were either constructed with a specific greedy algorithm or generated by assigning random values to the decision quantities. For all local search techniques the stop criterion was a predefined amount of computation time. This was in order to simplify the comparison.

4.2.1 Repeated descent

The following two types of moves are available in the neighborhood function of repeated descent:

- **One exchange.** The value of the selected decision quantity is changed to a randomly chosen new value within the range of the decision quantity.

- **Swap or two exchange.** The values of two selected decision quantities are interchanged.

The type of move and the decision quantities that participate in the move are selected at random. We used first improvement as pivot rule because
for all the studied problems the neighborhood of the current solution was too large to examine in full at each iteration. According to Anderson [1996] this rule seems to be superior during a local search process.

### 4.2.2 Tabu search

We used the moves one exchange and swap in the neighborhood function for the tabu search. In the applied algorithm for tabu search, we used a fixed, predefined list of 10 items which reflected the most recently accepted moves. The one exchange is stored as the position of the decision quantity and the new value. The swap is stored as the pair of positions of the involved decision quantities. This way of implementing the tabu list was successfully applied by Skorin-Kapov [1990] for the QAP. The pivot rule best improvement was replaced by first improvement, because we found a very low convergence rate in the trail runs. We did not use intensification or diversification during the search. The tabu status of a move is overruled if the move results in the best object value so far in the search process.

### 4.2.3 Simulated annealing

For simulated annealing we used the same neighborhood function and pivot rule as for repeated descent. The initial value of the control parameter or temperature was determined after a random walk of 100 iterations. The initial temperature should accept 95% of the deteriorations. We applied the geometric cooling schedule of Kirkpatrick, Gelatt & Vecchi [1983] on the temperature. In this cooling schedule the number of cycles between the adjustment of the control parameter is fixed and equal to the number of decision quantities. At each adjustment, the control parameter was decreased by 5%. Alternative cooling schedules as described by [Hajek, 1988] and Aarts & Van Laarhoven [1985] did not find better solutions in trail runs and proved to be too elaborate for search processes with limited run-times. Studies on the quadratic assignment problem with simulated annealing are described by Burkard & Rendl [1984] and Bonomi & Lutton [1986].

### 4.2.4 Parallel descent

Parallel descent works with a population of solutions. We used a population of 50 current solutions. The size of the population remained fixed during the search process. The neighborhood function consists of the moves one exchange or mutation and the replacement operator, which
4.2. Implementation

changes more decision quantities at the same time.

The selection process runs in parallel with the moves of the neighborhood function. Every selection operation starts by taking a sample of five solutions from the population, which is equal to 10% of the total population. The solutions in the sample are ranked according to the individual cost of the solutions. The solution with the best cost in the sample replaces the worst solution in the sample that resembles the best solution most closely. This selection process based on the crowding model of De Jong [1975] ensures high diversity in the population.

The frequency of mutation relative to the selection process is 0.95 and the relative frequency of replacement is 0.05. We protect the best solution of the population against being destroyed by the moves in the neighborhood function. This is called an elitist strategy.

4.2.5 Genetic algorithms

Genetic algorithms use strings of bytes or chromosomes of fixed length. The decision quantities are represented by one or more bytes on the string. A chromosome corresponds with a solution. Genetic algorithms search the neighborhood with the following two operators.

- **Mutation.** One byte at a single position of a randomly selected chromosome is replaced with a random number in the range of the chosen alphabet. Mutation resembles one point change.

- **Crossover.** Two parent chromosomes are selected at random from the population. The chromosomes exchange parts of the string with a two-point crossover. Only the resulting child chromosomes return to the population.

Our implementation of the genetic algorithm used a fixed population of 50 chromosomes or solutions and the same selection process as the parallel descent technique.

As is pointed out in the research of Grefenstette [1986], finding the correct frequencies of the mutation and the crossover operator relative to the select process is an optimization problem on itself for a genetic algorithm. We circumvented this problem partly by making the frequencies dependent on the status of the solutions in the population. The frequencies at which the operators are carried out relative to the selection process is dynamically adjusted during the search process and depend on the variation in the population. Mutations introduce variation in the population. In
general, this frequency must be kept low to avoid too much disturbances in the search for better solutions. Crossover has less effect in a population with low variation. A high mutation frequency can introduce higher variation on all positions of the chromosomes. High variation restores a high crossover frequency. We determine the variation in the population by measuring the fraction of solutions that are equal to the best solution so far, denote by \( B \in (0, 1] \subset \mathbb{R} \). We made the mutation frequency equal to \( B \) and the crossover frequency equal to \( (1 - B) \).

4.3 Constraints

Problems such as APP are highly constrained. According to Goldberg [1989] and Mattfeld [1996], we can incorporate constraints in the search process by using one of the following methods:

- **Reject all infeasible solutions.** A single violation of the constraints results directly in an infeasible solution. This procedure is impractical for highly constrained problems, because finding a feasible solution is almost as difficult as finding the best.

- **Select a neighborhood function that satisfies the constraints.** A move from a feasible solution ends in a feasible solution. To be able to start the search, we need a construction algorithm that produces feasible solutions.

- **Penalty method.** The constrained problem is transformed to a unconstrained problem by associating a cost or penalty with all constraint violations. These costs are included in the object function. With the penalty method we obtain some information out of infeasible solutions.

We did not apply the rejection method, due to the high constrainedness of our problems. Whenever possible, we tried to find a construction algorithm that produces feasible solutions. This is often as difficult as finding a neighborhood function that satisfies the constraints. In the absence of a suitable neighborhood function and/or construction algorithm, we applied the penalty method to obtain a unconstrained problem.
The two previous chapters outlined the slot planning problem and the local search techniques needed to solve this problem. In this chapter we present a solution strategy for the assignment problem for perishables (APP). We start with some general aspects of solving the APP with local search; more specifically we present the handling of constraints, and construction algorithms for initial solutions. Next, we present computational studies of both the space oriented strategy and the handling oriented strategy, as outlined in Section 3.2.5. We also experimented with extensions of the APP that adjust the storage conditions and redesign the layout of the zones. We conclude this chapter with a discussion of the effects of a slot plan on the operational assignment in a distribution center for perishables.

5.1 Solution strategy for the APP

Because of the complexity of the APP, investigated in Section 3.2.4, we decided to use the local search techniques, introduced in Chapter 4, to find solutions of problem instances of the APP. The APP is a highly constrained problem, both in storage capacity and in keeping quality loss. We could not find a construction algorithm that produces only feasible solutions, neither did we find a neighborhood function that satisfies both the
capacity and the keeping quality loss constraints. Therefore, we transformed both versions of the APP, with and without fixed assignments throughout the year as introduced in Section 3.2.5, to a unconstrained version with the penalty method, described in Section 4.3.

We first formulate the local search version of the APP|V, denoted with APPL|V, which allows variable assignments of the quality change groups throughout the year.

**Definition 5.1 (APPL|V).** Given is an instance of the local search version of the assignment problem for Perishables with variable assignments throughout the year (APPL|V) by a 13-tuple PD = (Q, T, w_q, Z, V_z, ζ, κ, ε_p, Λ, Λ_max, ω_q, π_c, π_q), where π_c ∈ IR^+ is the penalty for exceeding the storage capacity and π_q ∈ IR^+ is the penalty for exceeding the maximum allowed keeping quality loss. The other sets and functions are described in Definition 3.36. The problem is to find an assignment for perishables x_z, as described in Definition 3.36, that minimizes the cost function

\[
C_{APPL|V} = \omega_q \sum_{q \in Q} \max_{t \in T} \Lambda(q, x_z(q,t), \varepsilon(x_z(q,t), t)) \\
+ \pi_q \cdot \sum_{q \in Q} \sum_{t \in T} \text{sign}(w_q(q,t)) \\
\cdot \max(0, \Lambda(q, x_z(q,t), \varepsilon(x_z(q,t), t)) - \Lambda_{\text{max}}(q)) \\
+ \pi_c \cdot \sum_{z \in Z} \sum_{t \in T} \max(0, \left(\sum_{q \in Q} x_z(q,t) = z \right) w_q(q,t) - V_z(z)),
\]

with

\[
\text{sign}(x) = \begin{cases} 
1 & \text{if } x > 0 \\
0 & \text{if } x = 0 \\
-1 & \text{if } x < 0.
\end{cases}
\]

and ε(z, t) ∈ IR^+ the ethylene concentration in zone z in planning period t, as described in (3.70).

By replacing in Definition 5.1 the function x_z : Q × T → Z with the function \( \hat{x}_z : Q → Z \), described in Section 3.2.5 as (3.71), we get the definition of the local search version of the APP for fixed assignments throughout the year, denoted by APPL|F.

The choice of the penalties π_c and π_q determines the priority between storage capacity and keeping quality loss. We have to choose the values of π_c and π_q in such a way that all solutions with a cost higher than π_c...
or $\pi_q$ can be identified as infeasible solutions for APP. The cost of all the feasible solutions of the APPL|V are equal to the cost of the corresponding solutions of the APPL|V. Therefore we can use the APPL|V for the local search techniques without loss of information for the solutions of APP|V. The same is true for the relation between the APPL|F and the APP|F.

The neighborhood function of a local search algorithm determines a set of candidate solutions that can be reached from the present solution by moves such as one exchanges and swaps. For the APPL|F, a move changes the assignment of a quality change group to a zone. In the case of variable assignments throughout the year, as described in the definition of the APPL|V, a move changes the assignment of a quality change group to a zone in a planning period. The set of moves implemented for the local search techniques repeated descent, tabu search, simulated annealing, parallel descent, and genetic algorithms result in a connected neighborhood for both the APPL|V and the APPL|F, i.e., the local search techniques can reach the optimal solution from a feasible initial solution. All initial solution obtained by randomly assigning quality change groups to zones are feasible solutions for the APPL. The details of the implementation of the five local search techniques are described in Section 4.2.

The product interactions make it difficult to check if a move will cause a violation of the keeping quality constraints. The effect of ethylene on the keeping quality is non-linear and can only be calculated with the knowledge of the exact concentration of that agent in that particular zone. According to (3.70) the removal or addition of a quality change group to a zone causes a change in the ethylene production and in the effective volume of the zone. Therefore, we recalculated the complete cost function after each move, regardless of the type of move.

5.2 Problem generation

In order to test the solution strategy for the APPL|F and the APPL|F, we created 75 test instances, based on real world data obtained from the wholesaler of vegetables and fruits, described as case example in Section 2.5.

Quality change groups. The articles in the assortment belong to 180 quality change groups, i.e., $|Q| = 180$. The size of the assortment was the same for all instances. The maximum allowed relative keeping quality loss $\Lambda_{\text{max}}(q)$ for each quality change group $q \in Q$ is set to 0.7. This relatively
large value was chosen to make feasible slot plans possible for warehouse layouts with only two zones.

Table 5.1: Zone capacities and storage conditions for different numbers of zones, used for the test instances of the APPL. The column ‘\( V_z(z) \)’ denotes the zone capacity, ‘\( \kappa(z) \)’ the ventilation rate, ‘\( \sigma(z) \)’ the accessibility, and ‘\( \tau(z) \)’ the temperature for each zone \( z \in \mathcal{Z} \).

| \(|\mathcal{Z}|\) | \(z\) | \( V_z(z) \) | \( \kappa(z) \) | \( \sigma(z) \) | \( \tau(z) \) |
|---|---|---|---|---|---|
| 2 | 1 | 700 | 50 | 1.00 | 8.0 |
| | 2 | 1100 | 1 | 0.80 | 0.0 |
| 3 | 1 | 350 | 50 | 1.00 | 10.0 |
| | 2 | 350 | 1 | 0.80 | 4.0 |
| | 3 | 1100 | 1 | 0.80 | 0.0 |
| 4 | 1 | 300 | 50 | 1.00 | 10.0 |
| | 2 | 200 | 1 | 0.80 | 8.0 |
| | 3 | 200 | 1 | 0.80 | 4.0 |
| | 4 | 1100 | 1 | 0.80 | 0.0 |
| 6 | 1 | 300 | 50 | 1.00 | 10.0 |
| | 2 | 100 | 1 | 0.80 | 8.0 |
| | 3 | 100 | 1 | 0.80 | 7.0 |
| | 4 | 100 | 1 | 0.80 | 4.0 |
| | 5 | 100 | 1 | 0.80 | 2.0 |
| | 6 | 1100 | 1 | 0.80 | 0.0 |
| 8 | 1 | 250 | 50 | 1.00 | 10.0 |
| | 2 | 75 | 1 | 0.80 | 9.0 |
| | 3 | 75 | 1 | 0.80 | 8.0 |
| | 4 | 75 | 1 | 0.80 | 7.0 |
| | 5 | 75 | 1 | 0.80 | 5.0 |
| | 6 | 75 | 1 | 0.80 | 4.0 |
| | 7 | 75 | 1 | 0.80 | 2.0 |
| | 8 | 1100 | 1 | 0.80 | 0.0 |

**Zones.** We created sets of zones with two, three, four, six, and eight different zones: \{ Z2, Z3, Z4, Z6, Z8 \}. We did not include a set with a single zone since the probability of finding a feasible solution for problem instances based on such a set is small. The conditions in a single zone are seldom satisfactory to all quality change groups. We chose different zone conditions depending on both the number of zones and the properties of the set of 180 quality change groups. The zone capacities and the storage conditions for the different number of zones are described in Table 5.1. The
column ‘$V_z(z)$’ gives the zone capacity of each zone $z \in \mathcal{Z}$. The column ‘$\kappa(z)$’ presents the ventilation rate, ‘$\sigma(z)$’ the accessibility, and ‘$\tau(z)$’ the storage temperature of each zone. The total volume of a zone is always taken equal to three times the storage capacity of the zone, i.e., $\zeta(z) = 3 \cdot V_z(z)$ for each zone $z \in \mathcal{Z}$. The total storage capacity of the distribution center $\hat{V} = \sum_{z \in \mathcal{Z}} V_z(z)$ was in all instances the same: 1800. All storage capacities are expressed in unit load volumes.

![Figure 5.1: The sum of the group storage requirements during a year of the datasets R, A, and B.](image)

**Stock levels.** We used three stock level datasets for the problem instances. Dataset R was directly based on the stock level data of the wholesaler. The stock levels in datasets A and B were created using a random process. The difference between dataset A and B is the application of different seeds for the random generator. We generated stock levels in datasets A and B in the following way. The stock level of a quality change group $\zeta$ is a function of two stochastic variables: $u_1$ and $u_2$. We selected $u_1$ from a continuous uniform distribution on the interval $(0, 1] \subset \mathbb{R}$ and $u_2$ from a continuous uniform distribution on the interval
(1, 10] ⊆ \mathbb{R}. We can now calculate the stock level as

$$\zeta = \begin{cases} 
0 & \text{if } u_1 < P(\zeta = 0) \\
[u_2] & \text{otherwise.} 
\end{cases} \quad (5.2)$$

Around 20% of the year, a quality change group is not in stock and the stock level is zero, i.e. \( P(\zeta = 0) = 0.2 \). The average aggregated stock level of all the quality change groups is 694 in dataset R. For dataset A and B, the average aggregated stock level generated with (5.2) is 720. We chose the group storage space requirement \( w_q(q, t) \) equal to two times the average stock level of quality change group \( q \in Q \) in planning period \( t \in T \). Therefore, \( w_q(q, t) \) depends on the number of planning periods in the instance.

Based on datasets R, A, and B, we created sets of stock levels with one, two, three, four, and six planning periods: \{ A1, A2, \ldots, R4, R6 \}. More than six planning periods result in more revisions in the slot plan than can be implemented in the distribution center. Since a year can be divided in 12 months or 52 weeks, five planning periods do not result in a practical period length. Therefore, we did not consider instances with five planning periods.

As can be seen in Figure 5.1, the average of the sum of the group storage space requirements was approximately 80% of the total storage capacity of the distribution center. In this way, the test instances that we generated are highly capacitated. Martello & Toth [1991] showed that comparable generalized assignment problems are difficult from a computational point of view.

We used the following values for the weightfactor and penalties in all the instances: \( \varpi_q = 10^4 / |Q| \), \( \pi_c = 10^{10} \) and \( \pi_q = 10^7 \). The value of \( \pi_c \) is higher than \( \pi_q \) because we give a higher priority to the storage capacity constraints than to the maximum allowed keeping quality loss constraints. The cost of a feasible solution is always less or equal to \( |Q| \cdot \varpi_q = 10^4 \). The cost of an infeasible solution is greater or equal than \( \min(\pi_c, \pi_q) = 10^7 \). The large difference between \( \pi_c \) and \( \pi_q \) makes it possible to distinguish capacity and quality loss violations when examining the infeasible solutions.

Each of the 75 test instances is a combination of a set of zones and a set of stock levels. The set of stock levels includes the number of planning periods. For example, the problem instance denoted with Z6A2 has six zones and uses the stock level set A2 with two planning periods.
5.3 Initial solutions and lower bounds

In the case that the storage space capacity is not exceeded, and each quality change group can be stored at optimal conditions, the value of the cost of the APPL given in (5.1) is equal to zero. This also holds for non-perishables. For perishables, we can construct such an optimal solution by assigning each quality change group to a separate zone with enough storage capacity for each planning period. This requires a relatively large number of zones and too much storage capacity. In practice, we have a limited number of zones, so that quality change groups often have to share the same zone.

We need a construction algorithm for the APPL that generates a feasible solution for the majority of the problem instances. We observed that constructing an initial solution by assigning the quality change groups to random zones does seldom lead to a feasible solution. We propose two different greedy algorithms, both based on the Best Fit algorithm for the bin packing problem as described by Garey & Johnson [1979]. The proposed greedy algorithms focus on the keeping quality constraints for the selecting of the ‘best’ zone by indexing the zones on potential relative keeping quality loss for each quality change group that has to be assigned. For the calculation of the potential relative keeping quality loss, we ignore the product interactions by assuming that the ethylene concentration is very small \( \varepsilon(z,t) = \varepsilon_{\text{low}} \ll 1 \) for each zone \( z \in Z \) in planning period \( t \in T \). Violations of the storage capacity restrictions are temporary resolved by assigning the quality change group to the ‘worst’ zone, i.e., the last zone that was considered for assignment. After applying the construction algorithm, we have to recalculate the cost function to check whether the storage capacity was sufficient and whether the assumption about the product interaction was valid, and hence the constructed initial solution is feasible.

The first greedy algorithm, called the Best Fit Quality algorithm for variable assignments (BFQUALVAR), assigns quality change groups to zones for each planning period separately.

---

**Best Fit Quality algorithm for variable assignments (BFQUALVAR)**

1. Let \( T := |T| \), \( Q := |Q| \), and \( Z := |Z| \). Start with the first planning period, i.e., \( t := 1 \).
2. Sort the quality change groups $q_1, \ldots, q_Q$ such that
\[ w_q(q_{\phi(1)}, t) \geq \cdots \geq w_q(q_{\phi(Q)}, t), \]
with $\phi : \mathbb{N} \rightarrow \mathbb{N}$. Let for all $z \in Z$, $\bar{V}(z) := V_z(z)$. Start with the largest group by setting $i := 1$.

3. Sort the zones $z_1, \ldots, z_Z$ such that
\[ \Lambda(q_{\phi(i)}, z_{\varphi(1)}, \varepsilon_{\text{low}}) \leq \cdots \leq \Lambda(q_{\phi(i)}, z_{\varphi(Z)}, \varepsilon_{\text{low}}), \]
with $\varphi : \mathbb{N} \rightarrow \mathbb{N}$. Start with the zone with the lowest keeping quality loss by setting $j := 1$.

4. While $j < Z$ and $\bar{V}(z_{\varphi(j)}) - w_q(q_{\phi(i)}, t) < 0$ do $j := j + 1$.

5. Assign the quality change group to the zone: $x_z(q_{\phi(i)}, t) := z_{\varphi(j)}$ and $\bar{V}(z_{\varphi(j)}) := \bar{V}(z_{\varphi(j)}) - w_q(q_{\phi(i)}, t)$. When the quality change group does not fit in any of the zones, we assign the group to the last zone in the list.

6. $i := i + 1$. If $i \leq Q$, then go to 3.

7. $t := t + 1$. If $t \leq T$, then go to 2. Otherwise stop.

The BFQUALVAR is a greedy algorithm that assigns the quality change groups one by one, in a given order, to the zones. To establish the order of the assignments, we sort in Step 2 the quality change groups in non-increasing order on the group storage space requirement $w_q(q, t)$. Before we start the assignments in a planning period, we set the remaining storage capacity $\bar{V}(z)$ equal to the zone capacity $V_z(z)$. We start the assignment with the largest groups because these groups require the largest amount of storage capacity. For each quality change group, we sort in Step 3 the zones in non-decreasing order on the expected amount of relative keeping quality loss. We try to fit the quality change group in the zone where we expect the lowest amount of relative keeping quality loss, and where there is still enough remaining storage capacity $\bar{V}(z)$. When we do not find a zone with enough remaining storage capacity, we assign in Step 5 the quality change group to the last zone of the list. This zone has the worst storage conditions for the quality change group. We expect that this strategy increases the penalty cost for keeping quality loss but can sometimes avoid an increase in the penalty cost for storage capacity. The assignment process in Step 2 to 6 is repeated for each planning period. After the construction of an initial solution with BFQUALVAR, we
calculate the cost function of APPL. We have found an infeasible solution for APP when the cost exceeds the value of \( \pi_q \) or \( \pi_c \). We now investigate the time complexity of BFQUALVAR.

**Theorem 5.1.** BFQUALVAR finds an initial solution for the APPL in \( O(T \cdot Q \cdot \log Q + T \cdot Q \cdot Z \cdot \log Z) \) time.

**Proof.** Step 2 requires \( O(Q \cdot T) \) time for determining the parameters \( w_q(q,t) \), \( O(Q \cdot \log Q) \) time for indexing the variables and \( O(Z) \) time for initializing \( \tilde{V}(z) \). Calculating the expected relative keeping quality loss \( \Lambda \) in Step 2 and sorting the zones requires \( O(Z \cdot \log Z) \) time. Selecting a zone requires at most \( O(Z) \) time in Step 3. Assigning the quality change group in step 5 can be performed in constant time. Steps 3 to 5 are repeated \( T \cdot Q \) times and Step 2 is repeated \( T \) times. Hence, the time complexity BFQUALVAR is \( O(T \cdot Q \cdot \log Q + T \cdot Q \cdot Z \cdot \log Z) \).

In the case of fixed assignments throughout the year, we assign a quality change group to a zone for all planning periods at the same time. The construction algorithm for fixed assignments throughout the year is called BFQUALFIX.

---

**Best Fit Quality algorithm for fixed assignments (BFQUALFIX)**

1. Let \( Q := |Q| \) and \( Z := |Z| \). Let for all \( z \in Z \) and \( t \in T \) \( \tilde{V}(z,t) := V_z(z) \). Sort the quality change groups \( q_1, \ldots, q_Q \), such that

\[
\sum_{t \in T} w_q(q_{\phi(1)}, t) \geq \cdots \geq \sum_{t \in T} w_q(q_{\phi(Q)}, t),
\]

with \( \phi : \mathbb{N} \to \mathbb{N} \). Start with the largest group by setting \( i := 1 \).

2. Sort the zones \( z_1, \ldots, z_Z \) such that

\[
\Lambda(q_{\phi(i)}, z_{\phi(1)}, \varepsilon_{\text{low}}) \leq \cdots \leq \Lambda(q_{\phi(i)}, z_{\phi(Z)}, \varepsilon_{\text{low}}),
\]

with \( \varphi : \mathbb{N} \to \mathbb{N} \). Start with the zone with the lowest keeping quality loss by setting \( j := 1 \).

3. While \( j < Z \) and not for all \( t \in T \) : \( \tilde{V}(z_{\varphi(j)}, t) - w_q(q_{\phi(i)}, t) \geq 0 \) do \( j := j + 1 \).

4. Assign the quality change group to the zone: \( \hat{x}_z(q_{\phi(i)}) := z_{\varphi(j)} \) and for all \( t \in T \) : \( \tilde{V}(z_{\varphi(j)}, t) := \tilde{V}(z_{\varphi(j)}, t) - w_q(q_{\phi(i)}, t) \). When the quality change group does not fit in any of the zones, we assign the group to the last zone in the list.
5. \( i := i + 1 \). If \( i \leq Q \), then go to 2. Otherwise stop.

The main difference between \texttt{BFQUALVAR} and \texttt{BFQUALFIX} is establishing the order of the quality change groups for the greedy assignment process. In the \texttt{BFQUALVAR}, we have to sort the quality change groups for all planning periods at the same time. We chose for the sum of the group storage space requirements of each planning period but an good alternative is the maximum of the group storage space requirements of each planning period. This is discussed in Chapter 7.

**Theorem 5.2.** \texttt{BFQUALFIX} finds an initial solution for the \textsc{appl} in \( O(Q \cdot \log Q + Q \cdot Z \cdot \log Z + T \cdot Q \cdot Z) \) time.

**Proof.** Step 1 requires \( O(Q \cdot T) \) time for calculating the sums of \( w_q(q, t) \), \( O(Q \cdot \log Q) \) time for indexing the sums and \( O(Z \cdot T) \) time for initializing \( \hat{V}(z,t) \). Indexing the zones requires \( O(Z \cdot \log Z) \) time in Step 2 and finding a suitable zone takes at most \( O(Z \cdot T) \) time. Assigning the quality change group in step 4 can be performed in \( O(T) \) time. Steps 2 to 4 are repeated \( Q \) times. Hence, the time complexity of the \texttt{BFQUALFIX} algorithm is \( O(Q \cdot \log Q + Q \cdot Z \cdot \log Z + T \cdot Q \cdot Z) \). \hfill \Box

The time complexity of \texttt{BFQUALFIX} compared with that of \texttt{BFQUALVAR} is larger when only a few planning periods are required. However it gets smaller for a large number of planning periods. This is caused due to the assignment of a zone to a quality change group for several planning periods at the same time. Finally, we remark that in real world situations, the instances correspond with a relative small number of planning periods and zones compared to the number of quality change groups. Then, the time complexity of all the three greedy algorithms reduces to \( O(Q \cdot \log Q) \).

We can calculate the following lower bound when we set \( \pi_q \) and \( \pi_e \) equal to 0 in (5.1).

**Definition 5.2** (Lower bound keeping quality loss). A lower bound on the keeping quality loss is given by,

\[
\text{LB}_q = \omega_q \sum_{q \in \mathcal{Q}} \min_{z \in \mathcal{Z}} \Lambda(q, z, \varepsilon(z, t)),
\]

(5.3)

where the ethylene concentration \( \varepsilon(z, t) \ll 1 \) for all \( z \in \mathcal{Z} \) and \( t \in \mathcal{T} \), and \( \Lambda(q, z, \varepsilon(z, t)) \in \mathbb{R}_0^+ \) is the relative keeping quality loss of quality change group \( q \in \mathcal{Q} \) in zone \( z \). \hfill \Box
5.3. Initial solutions and lower bounds

This lower bound depends on the predefined zone conditions in the warehouse. Since the cost function deals with the maximum over all the planning periods, the bound $LB_q$ holds for both variable and fixed assignments throughout the year. The value of the bound $LB_q$ for the different sets of zones used in the problem instances of APPL is presented in Table 5.2. It can be seen that the lower bound decreases with the number of zones. A solution of APPL|V or APPL|F with a cost equal to the lower bound is optimal with respect to the zone conditions.

We used computational studies to investigate which of the above presented construction algorithms results in the largest number of feasible solutions. The results of the construction algorithms are shown in Tables 5.3 - 5.5. Each run of the construction algorithms took less than a second on a PentiumPro processor running under MS-DOS. An infeasible solution is denoted by 'Inf'. The cost of the best solution is presented in bold type and a solution with a cost equal to the lower bound is marked with an asterisk (*). These conventions are used for all tables with computational results in this thesis.

The algorithms BFQUALVAR and BFQUALFIX yield very slightly different results only. It appears in Table 5.3 that BFQUALVAR is the better algorithm for instances based on dataset A and that BFQUALFIX is the superior algorithm for dataset B, as in shown in Table 5.4. According to Table 5.5, BFQUALVAR gives better initial solutions for eight instances based on dataset R but constructs no feasible solution for an instance where BFQUALFIX succeeds in finding one. Closer inspection of the constructed solutions reveals that all infeasibilities were caused by the violation of the constraint on the maximum allowed relative keeping quality loss. One would expect that the BFQUALVAR with less restrictions on the

![Table 5.2: The calculated lower bounds for the APPL for the 75 test instances as a function of the applied set of zones. The different sets of zones are described in Table 5.1.](image)
Table 5.3: Results of the construction algorithms BFQUALVAR and BFQUALFIX for problem instances of the APPL based on dataset A.

<table>
<thead>
<tr>
<th></th>
<th>LBq</th>
<th>BFQUALVAR</th>
<th>BFQUALFIX</th>
</tr>
</thead>
<tbody>
<tr>
<td>Z2A1</td>
<td>838</td>
<td>838*</td>
<td>838*</td>
</tr>
<tr>
<td>Z2A2</td>
<td>838</td>
<td>838*</td>
<td>838*</td>
</tr>
<tr>
<td>Z2A3</td>
<td>838</td>
<td>838*</td>
<td>838*</td>
</tr>
<tr>
<td>Z2A4</td>
<td>838</td>
<td>838*</td>
<td>838*</td>
</tr>
<tr>
<td>Z2A6</td>
<td>838</td>
<td>838*</td>
<td>838*</td>
</tr>
<tr>
<td>Z3A1</td>
<td>698</td>
<td>Inf</td>
<td>Inf</td>
</tr>
<tr>
<td>Z3A2</td>
<td>698</td>
<td>Inf</td>
<td>Inf</td>
</tr>
<tr>
<td>Z3A3</td>
<td>698</td>
<td>Inf</td>
<td>Inf</td>
</tr>
<tr>
<td>Z3A4</td>
<td>698</td>
<td>Inf</td>
<td>Inf</td>
</tr>
<tr>
<td>Z3A6</td>
<td>698</td>
<td>Inf</td>
<td>Inf</td>
</tr>
<tr>
<td>Z4A1</td>
<td>485</td>
<td>485*</td>
<td>485*</td>
</tr>
<tr>
<td>Z4A2</td>
<td>485</td>
<td>485*</td>
<td>485*</td>
</tr>
<tr>
<td>Z4A3</td>
<td>485</td>
<td>499</td>
<td>504</td>
</tr>
<tr>
<td>Z4A4</td>
<td>485</td>
<td>515</td>
<td>525</td>
</tr>
<tr>
<td>Z4A6</td>
<td>485</td>
<td>515</td>
<td>520</td>
</tr>
<tr>
<td>Z6A1</td>
<td>413</td>
<td>434</td>
<td>434</td>
</tr>
<tr>
<td>Z6A2</td>
<td>413</td>
<td>437</td>
<td>446</td>
</tr>
<tr>
<td>Z6A3</td>
<td>413</td>
<td>441</td>
<td>487</td>
</tr>
<tr>
<td>Z6A4</td>
<td>413</td>
<td>445</td>
<td>514</td>
</tr>
<tr>
<td>Z6A6</td>
<td>413</td>
<td>437</td>
<td>480</td>
</tr>
<tr>
<td>Z8A1</td>
<td>367</td>
<td>Inf</td>
<td>Inf</td>
</tr>
<tr>
<td>Z8A2</td>
<td>367</td>
<td>Inf</td>
<td>Inf</td>
</tr>
<tr>
<td>Z8A3</td>
<td>367</td>
<td>Inf</td>
<td>Inf</td>
</tr>
<tr>
<td>Z8A4</td>
<td>367</td>
<td>Inf</td>
<td>Inf</td>
</tr>
<tr>
<td>Z8A6</td>
<td>367</td>
<td>Inf</td>
<td>Inf</td>
</tr>
</tbody>
</table>

Assignment of the quality change groups results in better solutions than the BFQUALFIX. The differences in the cost that we see between the solutions of the two algorithms can be attributed to the use of a potential relative keeping quality loss that ignored the effect of ethylene. The potential relative keeping quality loss can be quite different from the actual relative keeping quality loss when we assign an ethylene sensitive quality change group to a zone which is also occupied by ethylene producing quality change groups.

Since the solutions found by BFQUALFIX can be used to start the local search for both the APPL|F and the APPL|V and because we found only small differences in solution quality, we decided to use BFQUALFIX as the
5.4. Computational results

Table 5.4: Results of the construction algorithms BFQUALVAR and BFQUALFIX for problem instances of the APPL based on dataset B.

<table>
<thead>
<tr>
<th></th>
<th>LB₀</th>
<th>BFQUALVAR</th>
<th>BFQUALFIX</th>
</tr>
</thead>
<tbody>
<tr>
<td>Z2B1</td>
<td>838</td>
<td>838*</td>
<td>838*</td>
</tr>
<tr>
<td>Z2B2</td>
<td>838</td>
<td>838*</td>
<td>838*</td>
</tr>
<tr>
<td>Z2B3</td>
<td>838</td>
<td>838*</td>
<td>838*</td>
</tr>
<tr>
<td>Z2B4</td>
<td>838</td>
<td>838*</td>
<td>838*</td>
</tr>
<tr>
<td>Z2B6</td>
<td>838</td>
<td>838*</td>
<td>838*</td>
</tr>
<tr>
<td>Z3B1</td>
<td>698</td>
<td>Inf</td>
<td>Inf</td>
</tr>
<tr>
<td>Z3B2</td>
<td>698</td>
<td>Inf</td>
<td>Inf</td>
</tr>
<tr>
<td>Z3B3</td>
<td>698</td>
<td>Inf</td>
<td>Inf</td>
</tr>
<tr>
<td>Z3B4</td>
<td>698</td>
<td>Inf</td>
<td>Inf</td>
</tr>
<tr>
<td>Z3B6</td>
<td>698</td>
<td>Inf</td>
<td>Inf</td>
</tr>
<tr>
<td>Z4B1</td>
<td>485</td>
<td>485*</td>
<td>485*</td>
</tr>
<tr>
<td>Z4B2</td>
<td>485</td>
<td>485*</td>
<td>485*</td>
</tr>
<tr>
<td>Z4B3</td>
<td>485</td>
<td>485*</td>
<td>485*</td>
</tr>
<tr>
<td>Z4B4</td>
<td>485</td>
<td>526</td>
<td>511</td>
</tr>
<tr>
<td>Z4B6</td>
<td>485</td>
<td>523</td>
<td>511</td>
</tr>
<tr>
<td>Z6B1</td>
<td>413</td>
<td>436</td>
<td>436</td>
</tr>
<tr>
<td>Z6B2</td>
<td>413</td>
<td>451</td>
<td>436</td>
</tr>
<tr>
<td>Z6B3</td>
<td>413</td>
<td>474</td>
<td>432</td>
</tr>
<tr>
<td>Z6B4</td>
<td>413</td>
<td>470</td>
<td>444</td>
</tr>
<tr>
<td>Z6B6</td>
<td>413</td>
<td>481</td>
<td>467</td>
</tr>
<tr>
<td>Z8B1</td>
<td>367</td>
<td>Inf</td>
<td>Inf</td>
</tr>
<tr>
<td>Z8B2</td>
<td>367</td>
<td>Inf</td>
<td>Inf</td>
</tr>
<tr>
<td>Z8B3</td>
<td>367</td>
<td>Inf</td>
<td>Inf</td>
</tr>
<tr>
<td>Z8B4</td>
<td>367</td>
<td>Inf</td>
<td>Inf</td>
</tr>
<tr>
<td>Z8B6</td>
<td>367</td>
<td>Inf</td>
<td>Inf</td>
</tr>
</tbody>
</table>

construction algorithm for initial solutions in all performance tests in the next sections.

5.4 Computational results of the APPL

We carried out experiments for both versions of the APPL: the APPL/V with periodic reassignments of the quality change groups, and the APPL/F with fixed assignments throughout the year. The APPL/V is used for the space oriented decomposition strategy of the WS3P, and the APPL/F in the handling oriented strategy. The initial solutions found with BFQUALVAR and BFQUALFIX in Tables 5.3 - 5.5 did not show a clear
Table 5.5: Results of the construction algorithms BFQUALVAR and BFQUALFIX for problem instances of the APPL based on dataset R.

<table>
<thead>
<tr>
<th></th>
<th>LBₙ</th>
<th>BFQUALVAR</th>
<th>BFQUALFIX</th>
</tr>
</thead>
<tbody>
<tr>
<td>Z2R1</td>
<td>838</td>
<td>838*</td>
<td>838*</td>
</tr>
<tr>
<td>Z2R2</td>
<td>838</td>
<td>838*</td>
<td>838*</td>
</tr>
<tr>
<td>Z2R3</td>
<td>838</td>
<td>838*</td>
<td>838*</td>
</tr>
<tr>
<td>Z2R4</td>
<td>838</td>
<td>Inf</td>
<td>Inf</td>
</tr>
<tr>
<td>Z2R6</td>
<td>838</td>
<td>Inf</td>
<td>Inf</td>
</tr>
<tr>
<td>Z3R1</td>
<td>698</td>
<td>840</td>
<td>851</td>
</tr>
<tr>
<td>Z3R2</td>
<td>698</td>
<td>Inf</td>
<td>Inf</td>
</tr>
<tr>
<td>Z3R3</td>
<td>698</td>
<td>Inf</td>
<td>Inf</td>
</tr>
<tr>
<td>Z3R4</td>
<td>698</td>
<td>Inf</td>
<td>Inf</td>
</tr>
<tr>
<td>Z3R6</td>
<td>698</td>
<td>Inf</td>
<td>Inf</td>
</tr>
<tr>
<td>Z4R1</td>
<td>485</td>
<td>485*</td>
<td>485*</td>
</tr>
<tr>
<td>Z4R2</td>
<td>485</td>
<td>485*</td>
<td>485*</td>
</tr>
<tr>
<td>Z4R3</td>
<td>485</td>
<td>485*</td>
<td>485*</td>
</tr>
<tr>
<td>Z4R4</td>
<td>485</td>
<td>631</td>
<td>666</td>
</tr>
<tr>
<td>Z4R6</td>
<td>485</td>
<td>582</td>
<td>586</td>
</tr>
<tr>
<td>Z6R1</td>
<td>413</td>
<td>413*</td>
<td>460</td>
</tr>
<tr>
<td>Z6R2</td>
<td>413</td>
<td>413*</td>
<td>413*</td>
</tr>
<tr>
<td>Z6R3</td>
<td>413</td>
<td>413*</td>
<td>413*</td>
</tr>
<tr>
<td>Z6R4</td>
<td>413</td>
<td>502</td>
<td>513</td>
</tr>
<tr>
<td>Z6R6</td>
<td>413</td>
<td>498</td>
<td>541</td>
</tr>
<tr>
<td>Z8R1</td>
<td>367</td>
<td>462</td>
<td>465</td>
</tr>
<tr>
<td>Z8R2</td>
<td>367</td>
<td>585</td>
<td>Inf</td>
</tr>
<tr>
<td>Z8R3</td>
<td>367</td>
<td>Inf</td>
<td>457</td>
</tr>
<tr>
<td>Z8R4</td>
<td>367</td>
<td>Inf</td>
<td>502</td>
</tr>
<tr>
<td>Z8R6</td>
<td>367</td>
<td>518</td>
<td>738</td>
</tr>
</tbody>
</table>

victory for one of the two strategies. We expect that local search with the APPL|V finds better solutions than the APPL|F, because the APPL|V has less constraints on the assignments.

5.4.1 Algorithmic performance

The results of the tests with repeated descent, tabu search, simulated annealing, parallel descent, and genetic algorithms are given in the Tables 5.6 - 5.15 for both the APPL|V and the APPL|F, and for each set of zones. For each problem instance, we performed five runs using different seeds for the random number generator. Each run of each local search technique started with an initial solution, which was constructed
with BFQUALFix. The column ‘Init’ gives the cost of the initial solution, ‘min’ the minimal cost, and ‘avg’ the average cost over five runs. The column ‘%inf’ reports the percentage of the runs that did not succeed in finding a feasible solution. Each run took a computation time of 600 seconds on a 200 MHz PentiumPro processor running under MS-DOS.

5.4.2 Discussion

When we discuss the results of the tests presented in the Tables 5.6 - 5.15, we focus on the difference between the handling oriented and the space oriented strategy, and the performance of the local search techniques. Besides investigating the relation between the cost of a solution and the keeping quality loss of the individual quality change groups, we also examine the effect of the number of zones and the number of planning periods on the solution quality.

**Decomposition strategy.** Each solution of the APPL|F is also a solution for the APPL|V. The APPL|F has more constraints on the assignment of perishables to zones than the APPL|V with the possibility of reassignments between the planning periods. We expected therefore that the APPL|V would result in better solutions than the APPL|F. When we compare the results in Tables 5.6 - 5.14 with the results in Tables 5.7 - 5.15, we find better slot plans with the APPL|F. We offer two possible explanations for these results. First, the number of possible decision values for any given problem instance is greater for the APPL|V than for the APPL|F. With a stop criterion for all local search techniques that is based on a fixed and therefore limited amount of computation time, this can be a disadvantage. Second, a seasonal quality change group does not have any effect on the keeping quality when it does not occur in a zone during a particular planning period. In such a case, reassigning of the quality change groups to avoid interactions is not necessary. Close inspection of the assignments in the solutions of the APPL|V reveals that the opportunity to change the assignments during the year is seldom used.

**Local search technique.** When we examine the results of the tests with the APPL|V in Tables 5.6 - 5.14, the parallel descent technique seems to perform well compared with the four other local search techniques, especially with more zones and more planning periods. In the computational tests with the APPL|F, presented in the Tables 5.7 - 5.15, we observe that the tabu search algorithm performed best in the majority of the runs and the genetic algorithm the worst. Both the parallel descent and the
Table 5.6: Results of the APPL/V with five different local search techniques for problem instances with two zones. The lower bound $LB_{ij}$ is equal to 838. The column ‘Init’ presents the initial solution for the problem instance, the column ‘min’ the minimal cost, and the column ‘avg’ presents the average cost over five runs. The column ‘%inf’ reports the percentage of the runs that did not succeed in finding a feasible solution.

<table>
<thead>
<tr>
<th></th>
<th>Init</th>
<th>Repeated descent</th>
<th>Tabu search</th>
<th>Simulated annealing</th>
<th>Parallel descent</th>
<th>Genetic algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td>Z2R1</td>
<td>838*</td>
<td>838* 838*</td>
<td>838* 838*</td>
<td>838* 838*</td>
<td>838* 838*</td>
<td>838* 838* 0</td>
</tr>
<tr>
<td>Z2A2</td>
<td>838*</td>
<td>838* 838*</td>
<td>838* 838*</td>
<td>838* 838*</td>
<td>838* 838*</td>
<td>838* 838* 0</td>
</tr>
<tr>
<td>Z2R2</td>
<td>838*</td>
<td>838* 838*</td>
<td>838* 838*</td>
<td>838* 838*</td>
<td>838* 838*</td>
<td>838* 838* 0</td>
</tr>
<tr>
<td>Z2A3</td>
<td>838*</td>
<td>838* 838*</td>
<td>838* 838*</td>
<td>838* 838*</td>
<td>838* 838*</td>
<td>838* 838* 0</td>
</tr>
<tr>
<td>Z2B3</td>
<td>838*</td>
<td>838* 838*</td>
<td>838* 838*</td>
<td>838* 838*</td>
<td>838* 838*</td>
<td>838* 838* 0</td>
</tr>
<tr>
<td>Z2R3</td>
<td>838*</td>
<td>838* 838*</td>
<td>838* 838*</td>
<td>838* 838*</td>
<td>838* 838*</td>
<td>838* 838* 0</td>
</tr>
<tr>
<td>Z2A4</td>
<td>838*</td>
<td>838* 838*</td>
<td>838* 838*</td>
<td>838* 838*</td>
<td>838* 838*</td>
<td>838* 838* 0</td>
</tr>
<tr>
<td>Z2B4</td>
<td>838*</td>
<td>838* 838*</td>
<td>838* 838*</td>
<td>838* 838*</td>
<td>838* 838*</td>
<td>838* 838* 0</td>
</tr>
<tr>
<td>Z2R4</td>
<td>Inf</td>
<td>1581 1600</td>
<td>1487 1529</td>
<td>1533 1597</td>
<td>854 926</td>
<td>890 1064 0</td>
</tr>
<tr>
<td>Z2A6</td>
<td>838*</td>
<td>838* 838*</td>
<td>838* 838*</td>
<td>838* 838*</td>
<td>838* 838*</td>
<td>838* 838* 0</td>
</tr>
<tr>
<td>Z2B6</td>
<td>838*</td>
<td>838* 838*</td>
<td>838* 838*</td>
<td>838* 838*</td>
<td>838* 838*</td>
<td>838* 838* 0</td>
</tr>
<tr>
<td>Z2R6</td>
<td>Inf</td>
<td>1885 1916</td>
<td>1853 1881</td>
<td>1836 1879</td>
<td>908 984</td>
<td>1096 1234 20</td>
</tr>
</tbody>
</table>
Table 5.7: Results of the APPL|F with five different local search techniques for problem instances with two zones. The lower bound $LB_q$ is equal to 838. The meaning of the columns is the same as in Table 5.6.

<table>
<thead>
<tr>
<th>Column</th>
<th>Init</th>
<th>Repeated descent</th>
<th>Tabu search</th>
<th>Simulated annealing</th>
<th>Parallel descent</th>
<th>Genetic algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>min</td>
<td>avg</td>
<td>min</td>
<td>avg</td>
<td>min</td>
</tr>
<tr>
<td>Z2A1</td>
<td>838*</td>
<td>838*</td>
<td>838*</td>
<td>838*</td>
<td>838*</td>
<td>838*</td>
</tr>
<tr>
<td>Z2B1</td>
<td>838*</td>
<td>838*</td>
<td>838*</td>
<td>838*</td>
<td>838*</td>
<td>838*</td>
</tr>
<tr>
<td>Z2R1</td>
<td>838*</td>
<td>838*</td>
<td>838*</td>
<td>838*</td>
<td>838*</td>
<td>838*</td>
</tr>
<tr>
<td>Z2A2</td>
<td>838*</td>
<td>838*</td>
<td>838*</td>
<td>838*</td>
<td>838*</td>
<td>838*</td>
</tr>
<tr>
<td>Z2B2</td>
<td>838*</td>
<td>838*</td>
<td>838*</td>
<td>838*</td>
<td>838*</td>
<td>838*</td>
</tr>
<tr>
<td>Z2R2</td>
<td>838*</td>
<td>838*</td>
<td>838*</td>
<td>838*</td>
<td>838*</td>
<td>838*</td>
</tr>
<tr>
<td>Z2A3</td>
<td>838*</td>
<td>838*</td>
<td>838*</td>
<td>838*</td>
<td>838*</td>
<td>838*</td>
</tr>
<tr>
<td>Z2B3</td>
<td>838*</td>
<td>838*</td>
<td>838*</td>
<td>838*</td>
<td>838*</td>
<td>838*</td>
</tr>
<tr>
<td>Z2R3</td>
<td>838*</td>
<td>838*</td>
<td>838*</td>
<td>838*</td>
<td>838*</td>
<td>838*</td>
</tr>
<tr>
<td>Z2A4</td>
<td>838*</td>
<td>838*</td>
<td>838*</td>
<td>838*</td>
<td>838*</td>
<td>838*</td>
</tr>
<tr>
<td>Z2B4</td>
<td>838*</td>
<td>838*</td>
<td>838*</td>
<td>838*</td>
<td>838*</td>
<td>838*</td>
</tr>
<tr>
<td>Z2R4</td>
<td>Inf</td>
<td>853</td>
<td>853</td>
<td>853</td>
<td>854</td>
<td>854</td>
</tr>
<tr>
<td>Z2A6</td>
<td>838*</td>
<td>838*</td>
<td>838*</td>
<td>838*</td>
<td>838*</td>
<td>838*</td>
</tr>
<tr>
<td>Z2B6</td>
<td>838*</td>
<td>838*</td>
<td>838*</td>
<td>838*</td>
<td>838*</td>
<td>838*</td>
</tr>
<tr>
<td>Z2R6</td>
<td>Inf</td>
<td>853</td>
<td>853</td>
<td>853</td>
<td>854</td>
<td>855</td>
</tr>
</tbody>
</table>
Table 5.8: Results of the APPL|V with five different local search techniques for problem instances with three zones. The lower bound $LB_q$ is equal to 698. The meaning of the columns is the same as in Table 5.6.

<table>
<thead>
<tr>
<th>Init</th>
<th>Repeated descent</th>
<th>Tabu search</th>
<th>Simulated annealing</th>
<th>Parallel descent</th>
<th>Genetic algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>min</td>
<td>avg</td>
<td>min</td>
<td>avg</td>
<td>min</td>
</tr>
<tr>
<td>Z3A1</td>
<td>Inf</td>
<td>698*</td>
<td>698*</td>
<td>698*</td>
<td>698*</td>
</tr>
<tr>
<td>Z3B1</td>
<td>Inf</td>
<td>698*</td>
<td>698*</td>
<td>698*</td>
<td>698*</td>
</tr>
<tr>
<td>Z3R1</td>
<td>851</td>
<td>698*</td>
<td>698*</td>
<td>698*</td>
<td>698*</td>
</tr>
<tr>
<td>Z3A2</td>
<td>Inf</td>
<td>698*</td>
<td>725</td>
<td>698*</td>
<td>719</td>
</tr>
<tr>
<td>Z3B2</td>
<td>Inf</td>
<td>711</td>
<td>728</td>
<td>698*</td>
<td>715</td>
</tr>
<tr>
<td>Z3R2</td>
<td>Inf</td>
<td>698*</td>
<td>713</td>
<td>698*</td>
<td>709</td>
</tr>
<tr>
<td>Z3A3</td>
<td>Inf</td>
<td>788</td>
<td>797</td>
<td>751</td>
<td>765</td>
</tr>
<tr>
<td>Z3B3</td>
<td>Inf</td>
<td>743</td>
<td>769</td>
<td>714</td>
<td>753</td>
</tr>
<tr>
<td>Z3R3</td>
<td>Inf</td>
<td>698*</td>
<td>701</td>
<td>698*</td>
<td>701</td>
</tr>
<tr>
<td>Z3A4</td>
<td>Inf</td>
<td>764</td>
<td>821</td>
<td>764</td>
<td>810</td>
</tr>
<tr>
<td>Z3B4</td>
<td>Inf</td>
<td>791</td>
<td>836</td>
<td>766</td>
<td>825</td>
</tr>
<tr>
<td>Z3R4</td>
<td>Inf</td>
<td>821</td>
<td>831</td>
<td>807</td>
<td>825</td>
</tr>
<tr>
<td>Z3A6</td>
<td>Inf</td>
<td>849</td>
<td>915</td>
<td>849</td>
<td>901</td>
</tr>
<tr>
<td>Z3B6</td>
<td>Inf</td>
<td>917</td>
<td>956</td>
<td>905</td>
<td>950</td>
</tr>
<tr>
<td>Z3R6</td>
<td>Inf</td>
<td>724</td>
<td>741</td>
<td>724</td>
<td>742</td>
</tr>
</tbody>
</table>
Table 5.9: Results of the APPL/F with five different local search techniques for problem instances with three zones. The lower bound $LB_q$ is equal to 698. The meaning of the columns is the same as in Table 5.6.

<table>
<thead>
<tr>
<th></th>
<th>Init</th>
<th>Repeated descent</th>
<th>Tabu search</th>
<th>Simulated annealing</th>
<th>Parallel descent</th>
<th>Genetic algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>init</td>
<td>min</td>
<td>avg</td>
<td>min</td>
<td>avg</td>
<td>min</td>
</tr>
<tr>
<td>Z3A1</td>
<td>Inf</td>
<td>698*</td>
<td>698*</td>
<td>698*</td>
<td>698*</td>
<td>698*</td>
</tr>
<tr>
<td>Z3B1</td>
<td>Inf</td>
<td>698*</td>
<td>698*</td>
<td>698*</td>
<td>698*</td>
<td>698*</td>
</tr>
<tr>
<td>Z3R1</td>
<td>851</td>
<td>698*</td>
<td>698*</td>
<td>698*</td>
<td>698*</td>
<td>698*</td>
</tr>
<tr>
<td>Z3A2</td>
<td>Inf</td>
<td>698*</td>
<td>698*</td>
<td>698*</td>
<td>698*</td>
<td>698*</td>
</tr>
<tr>
<td>Z3B2</td>
<td>Inf</td>
<td>698*</td>
<td>698*</td>
<td>698*</td>
<td>698*</td>
<td>698*</td>
</tr>
<tr>
<td>Z3R2</td>
<td>Inf</td>
<td>698*</td>
<td>698*</td>
<td>698*</td>
<td>698*</td>
<td>698*</td>
</tr>
<tr>
<td>Z3A3</td>
<td>Inf</td>
<td>698*</td>
<td>698*</td>
<td>698*</td>
<td>698*</td>
<td>698*</td>
</tr>
<tr>
<td>Z3B3</td>
<td>Inf</td>
<td>698*</td>
<td>698*</td>
<td>698*</td>
<td>698*</td>
<td>698*</td>
</tr>
<tr>
<td>Z3R3</td>
<td>Inf</td>
<td>698*</td>
<td>698*</td>
<td>698*</td>
<td>698*</td>
<td>698*</td>
</tr>
<tr>
<td>Z3A4</td>
<td>Inf</td>
<td>698*</td>
<td>700</td>
<td>698*</td>
<td>698*</td>
<td>698*</td>
</tr>
<tr>
<td>Z3B4</td>
<td>Inf</td>
<td>702</td>
<td>703</td>
<td>702</td>
<td>702</td>
<td>702</td>
</tr>
<tr>
<td>Z3R4</td>
<td>Inf</td>
<td>698*</td>
<td>698*</td>
<td>698*</td>
<td>698*</td>
<td>698*</td>
</tr>
<tr>
<td>Z3A6</td>
<td>Inf</td>
<td>698*</td>
<td>700</td>
<td>698*</td>
<td>698*</td>
<td>698*</td>
</tr>
<tr>
<td>Z3B6</td>
<td>Inf</td>
<td>706</td>
<td>709</td>
<td>703</td>
<td>705</td>
<td>703</td>
</tr>
<tr>
<td>Z3R6</td>
<td>Inf</td>
<td>698*</td>
<td>698*</td>
<td>698*</td>
<td>698*</td>
<td>698*</td>
</tr>
</tbody>
</table>
Table 5.10: Results of the APPL\mid V with five different local search techniques for problem instances with four zones. The lower bound $LB_q$ is equal to 485. The meaning of the columns is the same as in Table 5.6.

<table>
<thead>
<tr>
<th>Init</th>
<th>Repeated descent</th>
<th>Tabu search</th>
<th>Simulated annealing</th>
<th>Parallel descent</th>
<th>Genetic algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>min</td>
<td>avg</td>
<td>min</td>
<td>avg</td>
<td>min</td>
</tr>
<tr>
<td>Z4A1</td>
<td>485*</td>
<td>485*</td>
<td>485*</td>
<td>485*</td>
<td>485*</td>
</tr>
<tr>
<td>Z4B1</td>
<td>485*</td>
<td>485*</td>
<td>485*</td>
<td>485*</td>
<td>485*</td>
</tr>
<tr>
<td>Z4R1</td>
<td>485*</td>
<td>485*</td>
<td>485*</td>
<td>485*</td>
<td>485*</td>
</tr>
<tr>
<td>Z4A2</td>
<td>485*</td>
<td>485*</td>
<td>485*</td>
<td>485*</td>
<td>485*</td>
</tr>
<tr>
<td>Z4B2</td>
<td>485*</td>
<td>485*</td>
<td>485*</td>
<td>485*</td>
<td>485*</td>
</tr>
<tr>
<td>Z4R2</td>
<td>485*</td>
<td>485*</td>
<td>485*</td>
<td>485*</td>
<td>485*</td>
</tr>
<tr>
<td>Z4A3</td>
<td>499</td>
<td>499</td>
<td>499</td>
<td>499</td>
<td>499</td>
</tr>
<tr>
<td>Z4B3</td>
<td>485*</td>
<td>485*</td>
<td>485*</td>
<td>485*</td>
<td>485*</td>
</tr>
<tr>
<td>Z4R3</td>
<td>485*</td>
<td>485*</td>
<td>485*</td>
<td>485*</td>
<td>485*</td>
</tr>
<tr>
<td>Z4A4</td>
<td>511</td>
<td>511</td>
<td>511</td>
<td>511</td>
<td>511</td>
</tr>
<tr>
<td>Z4B4</td>
<td>666</td>
<td>620</td>
<td>627</td>
<td>596</td>
<td>610</td>
</tr>
<tr>
<td>Z4R4</td>
<td>515</td>
<td>515</td>
<td>515</td>
<td>511</td>
<td>511</td>
</tr>
<tr>
<td>Z4A6</td>
<td>511</td>
<td>511</td>
<td>511</td>
<td>511</td>
<td>511</td>
</tr>
<tr>
<td>Z4B6</td>
<td>586</td>
<td>573</td>
<td>575</td>
<td>573</td>
<td>573</td>
</tr>
</tbody>
</table>

Assignment of perishables to zones.
Table 5.11: Results of the APPL/F with five different local search techniques for problem instances with four zones. The lower bound LB₃ is equal to 485. The meaning of the columns is the same as in Table 5.6.

<table>
<thead>
<tr>
<th>Init</th>
<th>Init</th>
<th>Repeated descent</th>
<th>Tabu search</th>
<th>Simulated annealing</th>
<th>Parallel descent</th>
<th>Genetic algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>min avg</td>
<td>min avg</td>
<td>min avg</td>
<td>min avg</td>
<td>min avg</td>
</tr>
<tr>
<td>Z4A1</td>
<td>485*</td>
<td>485* 485*</td>
<td>485* 485*</td>
<td>485* 485*</td>
<td>485* 485*</td>
<td>485* 485*</td>
</tr>
<tr>
<td>Z4B1</td>
<td>485*</td>
<td>485* 485*</td>
<td>485* 485*</td>
<td>485* 485*</td>
<td>485* 485*</td>
<td>485* 485*</td>
</tr>
<tr>
<td>Z4R1</td>
<td>485*</td>
<td>485* 485*</td>
<td>485* 485*</td>
<td>485* 485*</td>
<td>485* 485*</td>
<td>485* 485*</td>
</tr>
<tr>
<td>Z4A2</td>
<td>485*</td>
<td>485* 486</td>
<td>485* 485*</td>
<td>485* 485*</td>
<td>485* 485*</td>
<td>485* 485*</td>
</tr>
<tr>
<td>Z4B2</td>
<td>485*</td>
<td>485* 486</td>
<td>485* 485*</td>
<td>485* 485*</td>
<td>485* 485*</td>
<td>485* 485*</td>
</tr>
<tr>
<td>Z4R2</td>
<td>485*</td>
<td>485* 485*</td>
<td>485* 485*</td>
<td>485* 485*</td>
<td>485* 485*</td>
<td>485* 485*</td>
</tr>
<tr>
<td>Z4A3</td>
<td>499</td>
<td>485* 494</td>
<td>485* 485*</td>
<td>485* 485*</td>
<td>485* 485*</td>
<td>486 490</td>
</tr>
<tr>
<td>Z4B3</td>
<td>485*</td>
<td>485* 490</td>
<td>485* 485*</td>
<td>485* 485*</td>
<td>485* 485*</td>
<td>485* 485*</td>
</tr>
<tr>
<td>Z4R3</td>
<td>485*</td>
<td>485* 485*</td>
<td>485* 485*</td>
<td>485* 485*</td>
<td>485* 485*</td>
<td>485* 491</td>
</tr>
<tr>
<td>Z4A4</td>
<td>515</td>
<td>485* 493</td>
<td>485* 485*</td>
<td>485* 485*</td>
<td>485* 485*</td>
<td>485* 488</td>
</tr>
<tr>
<td>Z4B4</td>
<td>511</td>
<td>487 491</td>
<td>485* 487</td>
<td>487 487</td>
<td>485* 488</td>
<td>485* 487</td>
</tr>
<tr>
<td>Z4R4</td>
<td>666</td>
<td>485* 485*</td>
<td>485* 485*</td>
<td>485* 485*</td>
<td>485* 485*</td>
<td>485* 489</td>
</tr>
<tr>
<td>Z4A6</td>
<td>515</td>
<td>485* 491</td>
<td>485* 485*</td>
<td>485* 485*</td>
<td>486 487</td>
<td>485* 490</td>
</tr>
<tr>
<td>Z4B6</td>
<td>511</td>
<td>487 491</td>
<td>485* 487</td>
<td>487 487</td>
<td>485* 488</td>
<td>487 494</td>
</tr>
<tr>
<td>Z4R6</td>
<td>586</td>
<td>485* 485*</td>
<td>485* 486</td>
<td>485* 485*</td>
<td>485* 485*</td>
<td>490 497</td>
</tr>
</tbody>
</table>
Table 5.12: Results of the APPL|V with five different local search techniques for problem instances with six zones. The lower bound LB_q is equal to 413. The meaning of the columns is the same as in Table 5.6.

<table>
<thead>
<tr>
<th>Init</th>
<th>Repeated descent</th>
<th>Tabu search</th>
<th>Simulated annealing</th>
<th>Parallel descent</th>
<th>Genetic algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>min</td>
<td>avg</td>
<td>min</td>
<td>avg</td>
<td>min</td>
</tr>
<tr>
<td>Z6A1</td>
<td>434</td>
<td>420</td>
<td>420</td>
<td>422</td>
<td>417</td>
</tr>
<tr>
<td>Z6B1</td>
<td>436</td>
<td>425</td>
<td>425</td>
<td>425</td>
<td>419</td>
</tr>
<tr>
<td>Z6R1</td>
<td>460</td>
<td>413*</td>
<td>413*</td>
<td>413*</td>
<td>413*</td>
</tr>
<tr>
<td>Z6A2</td>
<td>437</td>
<td>437</td>
<td>429</td>
<td>437</td>
<td>437</td>
</tr>
<tr>
<td>Z6B2</td>
<td>436</td>
<td>436</td>
<td>432</td>
<td>437</td>
<td>436</td>
</tr>
<tr>
<td>Z6R2</td>
<td>413*</td>
<td>413*</td>
<td>413*</td>
<td>413*</td>
<td>413*</td>
</tr>
<tr>
<td>Z6A3</td>
<td>441</td>
<td>441</td>
<td>441</td>
<td>441</td>
<td>441</td>
</tr>
<tr>
<td>Z6B3</td>
<td>432</td>
<td>432</td>
<td>432</td>
<td>432</td>
<td>432</td>
</tr>
<tr>
<td>Z6R3</td>
<td>413*</td>
<td>413*</td>
<td>413*</td>
<td>413*</td>
<td>413*</td>
</tr>
<tr>
<td>Z6A4</td>
<td>445</td>
<td>445</td>
<td>445</td>
<td>445</td>
<td>438</td>
</tr>
<tr>
<td>Z6B4</td>
<td>444</td>
<td>444</td>
<td>444</td>
<td>444</td>
<td>444</td>
</tr>
<tr>
<td>Z6R4</td>
<td>513</td>
<td>485</td>
<td>480</td>
<td>485</td>
<td>415</td>
</tr>
<tr>
<td>Z6A6</td>
<td>437</td>
<td>436</td>
<td>436</td>
<td>436</td>
<td>436</td>
</tr>
<tr>
<td>Z6B6</td>
<td>467</td>
<td>467</td>
<td>467</td>
<td>467</td>
<td>467</td>
</tr>
<tr>
<td>Z6R6</td>
<td>541</td>
<td>532</td>
<td>532</td>
<td>533</td>
<td>482</td>
</tr>
</tbody>
</table>
Table 5.13: Results of the APPLIF with five different local search techniques for problem instances with six zones. The lower bound $LB_q$ is equal to 413. The meaning of the columns is the same as in Table 5.6.

<table>
<thead>
<tr>
<th>Init</th>
<th>Repeated descent</th>
<th>Tabu search</th>
<th>Simulated annealing</th>
<th>Parallel descent</th>
<th>Genetic algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>min avg</td>
<td>min avg</td>
<td>min avg</td>
<td>min avg</td>
<td>min avg</td>
</tr>
<tr>
<td>Z6A1</td>
<td>434 420 423</td>
<td>417 419</td>
<td>420 422</td>
<td>417 419</td>
<td>428 431</td>
</tr>
<tr>
<td>Z6B1</td>
<td>436 422 431</td>
<td>419 423</td>
<td>425 425</td>
<td>419 421</td>
<td>436 436</td>
</tr>
<tr>
<td>Z6R1</td>
<td>460 413* 413*</td>
<td>413* 413*</td>
<td>413* 413*</td>
<td>413* 413*</td>
<td>438 438</td>
</tr>
<tr>
<td>Z6A2</td>
<td>437 422 429</td>
<td>417 418</td>
<td>417 419</td>
<td>417 423</td>
<td>430 435</td>
</tr>
<tr>
<td>Z6B2</td>
<td>436 422 425</td>
<td>419 422</td>
<td>422 425</td>
<td>423 426</td>
<td>436 436</td>
</tr>
<tr>
<td>Z6R2</td>
<td>413* 413* 414</td>
<td>413* 413*</td>
<td>413* 413*</td>
<td>413* 413*</td>
<td>413* 413*</td>
</tr>
<tr>
<td>Z6A3</td>
<td>441 422 444</td>
<td>422 424</td>
<td>422 422</td>
<td>422 426</td>
<td>435 440</td>
</tr>
<tr>
<td>Z6B3</td>
<td>432 422 435</td>
<td>419 423</td>
<td>422 425</td>
<td>419 423</td>
<td>432 432</td>
</tr>
<tr>
<td>Z6R3</td>
<td>413* 413* 413*</td>
<td>413* 413*</td>
<td>413* 413*</td>
<td>413* 413*</td>
<td>413* 413*</td>
</tr>
<tr>
<td>Z6A4</td>
<td>445 423 436</td>
<td>423 427</td>
<td>424 427</td>
<td>431 432</td>
<td>436 438</td>
</tr>
<tr>
<td>Z6B4</td>
<td>444 434 447</td>
<td>427 430</td>
<td>430 432</td>
<td>430 433</td>
<td>444 444</td>
</tr>
<tr>
<td>Z6R4</td>
<td>513 415 420</td>
<td>413* 413*</td>
<td>413* 413*</td>
<td>415 415</td>
<td>417 423</td>
</tr>
<tr>
<td>Z6A6</td>
<td>437 423 428</td>
<td>423 425</td>
<td>424 427</td>
<td>425 427</td>
<td>432 432</td>
</tr>
<tr>
<td>Z6B6</td>
<td>467 429 437</td>
<td>427 430</td>
<td>428 430</td>
<td>452 464</td>
<td>448 456</td>
</tr>
<tr>
<td>Z6R6</td>
<td>541 424 430</td>
<td>420 426</td>
<td>429 434</td>
<td>459 465</td>
<td>455 476</td>
</tr>
</tbody>
</table>
Table 5.14: Results of the APPLV with five different local search techniques for problem instances with eight zones. The lower bound $L_B$ is equal to 367. The meaning of the columns is the same as in Table 5.6.

<table>
<thead>
<tr>
<th>Init</th>
<th>Simulated annealing</th>
<th>Tabu search</th>
<th>Genetic algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>min</td>
<td>avg</td>
<td>min</td>
</tr>
<tr>
<td>ZA1</td>
<td>381</td>
<td>410</td>
<td>376</td>
</tr>
<tr>
<td>ZB1</td>
<td>395</td>
<td>417</td>
<td>376</td>
</tr>
<tr>
<td>ZB1</td>
<td>405</td>
<td>367*</td>
<td>368</td>
</tr>
<tr>
<td>ZR1</td>
<td>455</td>
<td>455</td>
<td>455</td>
</tr>
<tr>
<td>ZA2</td>
<td>406</td>
<td>430</td>
<td>392</td>
</tr>
<tr>
<td>ZB2</td>
<td>415</td>
<td>442</td>
<td>415</td>
</tr>
<tr>
<td>ZB2</td>
<td>418</td>
<td>433</td>
<td>418</td>
</tr>
<tr>
<td>ZA3</td>
<td>484</td>
<td>496</td>
<td>459</td>
</tr>
<tr>
<td>ZB3</td>
<td>490</td>
<td>516</td>
<td>490</td>
</tr>
<tr>
<td>ZB3</td>
<td>493</td>
<td>517</td>
<td>493</td>
</tr>
<tr>
<td>ZR3</td>
<td>457</td>
<td>477</td>
<td>457</td>
</tr>
<tr>
<td>ZA4</td>
<td>519</td>
<td>550</td>
<td>518</td>
</tr>
<tr>
<td>ZB4</td>
<td>530</td>
<td>530</td>
<td>530</td>
</tr>
<tr>
<td>ZB4</td>
<td>535</td>
<td>535</td>
<td>535</td>
</tr>
<tr>
<td>ZR4</td>
<td>502</td>
<td>502</td>
<td>502</td>
</tr>
<tr>
<td>ZA5</td>
<td>519</td>
<td>557</td>
<td>519</td>
</tr>
<tr>
<td>ZB6</td>
<td>567</td>
<td>600</td>
<td>567</td>
</tr>
<tr>
<td>ZR6</td>
<td>738</td>
<td>684</td>
<td>738</td>
</tr>
</tbody>
</table>

Assignment of perishables to zones
Table 5.15: Results of the APPL|F with five different local search techniques for problem instances with eight zones. The lower bound LB₈ is equal to 367. The meaning of the columns is the same as in Table 5.6.

<table>
<thead>
<tr>
<th></th>
<th>Init</th>
<th>Repeated descent</th>
<th>Tabu search</th>
<th>Simulated annealing</th>
<th>Parallel descent</th>
<th>Genetic algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>min</td>
<td>avg</td>
<td>min</td>
<td>avg</td>
<td>min</td>
</tr>
<tr>
<td>Z8A1</td>
<td>Inf</td>
<td>382</td>
<td>384</td>
<td>374</td>
<td>376</td>
<td>377</td>
</tr>
<tr>
<td>Z8B1</td>
<td>Inf</td>
<td>378</td>
<td>394</td>
<td>376</td>
<td>382</td>
<td>376</td>
</tr>
<tr>
<td>Z8R1</td>
<td>465</td>
<td>367*</td>
<td>370</td>
<td>367*</td>
<td>368</td>
<td>367*</td>
</tr>
<tr>
<td>Z8A2</td>
<td>Inf</td>
<td>384</td>
<td>395</td>
<td>377</td>
<td>378</td>
<td>381</td>
</tr>
<tr>
<td>Z8B2</td>
<td>Inf</td>
<td>378</td>
<td>391</td>
<td>376</td>
<td>377</td>
<td>417</td>
</tr>
<tr>
<td>Z8R2</td>
<td>Inf</td>
<td>367*</td>
<td>374</td>
<td>367*</td>
<td>367*</td>
<td>371</td>
</tr>
<tr>
<td>Z8A3</td>
<td>Inf</td>
<td>394</td>
<td>397</td>
<td>388</td>
<td>390</td>
<td>382</td>
</tr>
<tr>
<td>Z8B3</td>
<td>Inf</td>
<td>389</td>
<td>395</td>
<td>377</td>
<td>379</td>
<td>381</td>
</tr>
<tr>
<td>Z8R3</td>
<td>457</td>
<td>381</td>
<td>396</td>
<td>367*</td>
<td>375</td>
<td>377</td>
</tr>
<tr>
<td>Z8A4</td>
<td>Inf</td>
<td>400</td>
<td>414</td>
<td>392</td>
<td>396</td>
<td>385</td>
</tr>
<tr>
<td>Z8B4</td>
<td>Inf</td>
<td>383</td>
<td>396</td>
<td>382</td>
<td>383</td>
<td>387</td>
</tr>
<tr>
<td>Z8R4</td>
<td>502</td>
<td>387</td>
<td>406</td>
<td>378</td>
<td>379</td>
<td>391</td>
</tr>
<tr>
<td>Z8A6</td>
<td>Inf</td>
<td>399</td>
<td>416</td>
<td>389</td>
<td>392</td>
<td>387</td>
</tr>
<tr>
<td>Z8B6</td>
<td>Inf</td>
<td>382</td>
<td>400</td>
<td>381</td>
<td>381</td>
<td>385</td>
</tr>
<tr>
<td>Z8R6</td>
<td>738</td>
<td>412</td>
<td>428</td>
<td>402</td>
<td>408</td>
<td>417</td>
</tr>
</tbody>
</table>
Assignment of perishables to zones

Figure 5.2: The distribution of the relative keeping quality loss over the quality change groups for the best solutions of four problem instances.
simulated annealing algorithm found the lowest cost for a large number of instances. The success of the parallel descent method compared to the tabu search or repeated descent method in these experiments with local search may be explained partly by the parallel nature of the search and partly by the replacement operator or move. Both aspects make it possible to switch quickly from one part of the search space to another, new part of the search space. This is especially useful when the search space is large and when there are multiple local optima.

**Keeping quality loss.** Remember that a cost of 0 signifies that the slot plan stores all quality change groups and thus all articles in the assortment at ideal conditions. Higher cost reveals something about the average storage conditions. Individual quality change groups may loose as much as \( \Lambda_{\text{max}} \), the maximum allowed relative keeping quality loss which was in all test instances 70% for all quality change groups. The distribution of \( \Lambda \) over the quality change groups in the best solutions of the instances Z2R1, Z2R6, Z8R1, and Z8R6 are plotted in Figure 5.2. The problem instance Z8R1 with eight zones and a single planning period has the smallest deviation from the average compared to the other three instances. The quality change group ‘banana’ is one of the quality change groups that is often found in the tail of the distribution. A keeping quality problem of banana which is caused by chilling injury becomes rapidly visible through gray yellow coloring of the skin. Banana is an important group measured in yearly turnover and added value.

**Number of zones.** We discuss the results of the APPL separately for each number of zones. A larger number of zones is only interesting for a distribution center for perishables when it reduces the keeping quality loss considerably, since additional zones are making the handling more difficult and also increase the chance that the incoming articles cannot be stored. Therefore, we used the best solutions with two zones as point of reference by the comparison of solutions with more than two zones.

- **Two zones.** For the instances with only two zones of both the sets A and B, the construction algorithm BFQUALFIX is already able to find the optimal solution. Only the two instances based on dataset R with two zones and more than three planning periods seem to be hard. In Table 5.6, we see that for the APPL|V the parallel descent finds the lowest cost for these instances. The APPL|F finds solutions close to the lower bound, according to Table 5.7.
Three zones. All the instances with three zones are difficult from a computational point of view since the BFQUALF1X never constructs a feasible initial solution. Eventually, all local search techniques with the exception of the genetic algorithm find feasible solutions. The parallel descent method is able to find optimal or the lowest cost for all instances with three zones. Table 5.9 shows that all searches with APPL|F result in better solutions compared to the results with APPL|V in Table 5.8. The optimal cost of the slot plan with three zones is 16% lower than the optimal cost of the slot plan with two zones.

Four zones. The optimal solutions for the studied instances with four zones and less than three planning periods can all be constructed with the BFQUALF1X. With the APPL|V, the instances based on dataset A and B with three or more planning periods could not be improved by the five local search techniques, according to Table 5.10. The parallel descent method finds with the APPL|V the best solutions for the instances based on dataset R. With the APPL|F, the solutions for the instances with three and more planning periods are better than with the APPL|V, and the tabu search results in optimal solutions for all problem instances. The optimal cost of a slot plan with four zones is 42% lower than an optimal cost of a slot plan with two zones.

Six zones. The situation is not so clear for the instances with six zones. For up to three planning periods most local search techniques are able to find the optimal solution for set R. However, for instances based on dataset A and B we cannot find solutions with a cost equal to the lower bound. As shown in Table 5.12, the performance of the local search technique with APPL|V depends on the instance without any clear winners. With APPL|F, it can be seen in Table 5.13 that tabu search finds the best solutions for all problem instances. The optimal cost of the slot plan with six zones is just 51% better than the optimal cost of a slot plan with only two zones.

Eight zones. With eight zones we have almost the same situation as with six zones. In the case of the APPL|V, the parallel descent method performs on average the best, according to Table 5.14, and performs second best for the instance with eight zones and a single planning period. As is shown in Table 5.15, the tabu search performs on average the best in the case of the APPL|F. Observe that the results with the APPL|F are considerably better than with the
5.5. **Problem extensions**

APPL\|V in the case of three planning periods and more. The improvement compared with the optimal cost of a slot plan with two zones equals 56 \%, which is the largest improvement level of these test sets. The optimal slot plan with eight zones is only slightly better than a optimal slot plan with six zones.

In our limited computational study, the problem instances with four zones result in slot plans that have the highest relative improvement in cost over slot plans for problem instances with only two zones.

**Number of planning periods.** The effect of the number of planning periods on the group storage space requirement is not incorporated in the problem instances used for the computational study. For the wholesaler of vegetables and fruits, the relation between the summed peak stock levels of the quality change groups and the number of planning periods is shown in Figure 2.8. The summed peak stock levels give only the worst case scenario for the utilization of the storage capacity. We assumed that the group storage space requirements are lower than the summed peak stock levels. We investigated in this research a scenario based on problem instances where the group storage space requirements are equal to two times the average stock levels instead of equal to the peak stock levels. For the three datasets that we investigated in this way, more planning periods do not result in significantly better slot plans as far as keeping quality is concerned. The effect of the number of planning periods on the utilization of the storage capacity in the absence of keeping quality effects is discussed in Chapter 7.

From the analysis of the assortment data, presented in Section 2.5, and interviews with the users we concluded that three planning periods per year coincides with the actual change of seasons of the majority of the quality change groups in the vegetables and fruits business.

For the remaining experiments with variable zone conditions in this chapter, we restricted ourselves to fixed assignments throughout the year and we used the problem instances with three planning periods. This reduces the set of problem instances for the remaining experiments from 75 to 15.

### 5.5 Computational results of extensions of the APPL|F

Zone conditions such as temperature can easily be changed within certain ranges in a distribution center for perishables. Changing the zone capacities is normally more difficult. We introduce two extensions of the
APPL/F that make it possible to adjust the zone temperature with or without changing the zone capacities.

5.5.1 Variable zone temperature

In this section, we study the effect on the quality of the solution of the APPL/F by adding the decision variable $\tau_z(z)$, introduced in (3.72). The function $\tau_z(z)$ replaces the temperature function $\tau(z)$ for zone $z \in Z$ that is used in the quality change models to calculate the relative keeping quality loss $\Lambda$. The extended problem is called the assignment problem for perishables with fixed assignments and variable zone temperatures and is denoted with APPL/F/T.

Local search implementation. A move in the neighborhood function of the APPL/F/T changes the assignment of a quality change group to a zone and/or the assignment of a temperature to a zone. An assignment of a temperature to a zone cannot be exchanged with an assignment of a quality change group to a zone and vice versa. The temperature that can be assigned to a zone can assume one of the following values: 0.0, 0.1, ..., 19.9, 20.0. The set of moves implemented for the local search techniques repeated descent, tabu search, simulated annealing, parallel descent, and genetic algorithms result for both the assignment of quality change groups and the assignment of temperatures in a connected neighborhood for the APPL/F/T, i.e., the local search techniques can reach the optimal solution from a feasible initial solution with the restriction of the predefined precision of the temperature. All initial solution obtained by randomly assigning quality change groups to zones are feasible solutions for the APPL/F/T. Further details of the implementation of the five local search techniques are described in Section 4.2.

Initial solutions and lower bounds. The construction of an initial solution with BFQUALFIX depends heavily on the expected keeping quality loss in the zones. We used in the BFQUALFIX the temperatures that are printed in the last column of Table 5.1. Since we study the same problem instances as in Section 5.4 with respect to the required stock levels, we may assume that all these problem instances are feasible with respect to the zone capacity. The lower bound $LB_q$, described in Definition 5.2, is invalid when we change the zone conditions such as temperature. We were not able to find a lower bound that is independent of the zone conditions.

Algorithmic performance. For the following experiments we used the 15 problem instances with three planning periods as described in Sec-
Table 5.16: Results of the APPL|F|T with five different local search techniques for problem instances with three planning periods. The meaning of the other columns is the same as in Table 5.6.

<table>
<thead>
<tr>
<th></th>
<th>Init</th>
<th>Repeated descent</th>
<th>Tabu search</th>
<th>Simulated annealing</th>
<th>Parallel descent</th>
<th>Genetic algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>min</td>
<td>avg</td>
<td>min</td>
<td>avg</td>
<td>min</td>
<td>avg</td>
</tr>
<tr>
<td>Z2A3</td>
<td>838</td>
<td>838</td>
<td>838</td>
<td>838</td>
<td>838</td>
<td>838</td>
</tr>
<tr>
<td>Z2B3</td>
<td>838</td>
<td>838</td>
<td>838</td>
<td>838</td>
<td>838</td>
<td>838</td>
</tr>
<tr>
<td>Z2R3</td>
<td>838</td>
<td>838</td>
<td>838</td>
<td>838</td>
<td>838</td>
<td>838</td>
</tr>
<tr>
<td>Z3A3</td>
<td>Inf</td>
<td>670</td>
<td>670</td>
<td>670</td>
<td>670</td>
<td>670</td>
</tr>
<tr>
<td>Z3B3</td>
<td>Inf</td>
<td>674</td>
<td>671</td>
<td>671</td>
<td>671</td>
<td>671</td>
</tr>
<tr>
<td>Z3R3</td>
<td>Inf</td>
<td>653</td>
<td>670</td>
<td>670</td>
<td>670</td>
<td>671</td>
</tr>
<tr>
<td>Z4A3</td>
<td>499</td>
<td>485</td>
<td>485</td>
<td>485</td>
<td>485</td>
<td>485</td>
</tr>
<tr>
<td>Z4B3</td>
<td>485</td>
<td>485</td>
<td>485</td>
<td>485</td>
<td>485</td>
<td>485</td>
</tr>
<tr>
<td>Z4R3</td>
<td>485</td>
<td>485</td>
<td>485</td>
<td>485</td>
<td>485</td>
<td>485</td>
</tr>
<tr>
<td>Z6A3</td>
<td>441</td>
<td>427</td>
<td>422</td>
<td>422</td>
<td>422</td>
<td>422</td>
</tr>
<tr>
<td>Z6B3</td>
<td>432</td>
<td>427</td>
<td>422</td>
<td>422</td>
<td>422</td>
<td>422</td>
</tr>
<tr>
<td>Z6R3</td>
<td>413</td>
<td>413</td>
<td>413</td>
<td>413</td>
<td>413</td>
<td>413</td>
</tr>
<tr>
<td>Z8A3</td>
<td>Inf</td>
<td>400</td>
<td>392</td>
<td>404</td>
<td>388</td>
<td>407</td>
</tr>
<tr>
<td>Z8B3</td>
<td>Inf</td>
<td>395</td>
<td>381</td>
<td>391</td>
<td>395</td>
<td>397</td>
</tr>
<tr>
<td>Z8R3</td>
<td>457</td>
<td>367</td>
<td>367</td>
<td>371</td>
<td>367</td>
<td>371</td>
</tr>
</tbody>
</table>

The results of the test with the different local search techniques for the APPL|F|T are given in Table 5.16. The meaning of the columns is the same as in Table 5.6. Like in the previous experiments, we performed for each problem instance five runs using different seeds for the random number generator. Each run took a computation time of 600 seconds for the APPL|F|T on a 200 MHz PentiumPro processor running under MS-DOS.

**Discussion.** It can be observed that the results in Table 5.16 show only a small improvement compared with the results for the APPL|F|. For the instances with two or four zones, there is no difference at all. The largest improvements - even below the previous but now invalid bounds - are for the instances with three zones. For the instances with three zones based on dataset A and B, this improvement is caused by lowering the temperature of the first zone from 10° C to 9° C. The result of the problem instance Z3R3 is showing that in the first zone the temperature is changed to 7.2° C and in the second zone from 4° C to 10° C, compared to the initial zone conditions described in Table 5.1.
5.5.2 Layout redesign

When we design a new building or in certain types of existing buildings where the storage rooms have removable walls, we can determine the required storage capacity after we have assigned the quality change groups to the zones. In this section, we study a solution strategy for the APPL|F|T with variable zone capacities. The assignment problem for perishables with fixed assignments and redesign of the layout is denoted with APPL|F|R.

The APPL|F|R is an extension the APPL|F|T with the function $V_z'(z)$ that allocates zone capacity to each zone $z \in Z$. This replacement of the zone capacity function $V_z(z)$ was introduced in Section 3.2.6. We developed a solution strategy based on a relaxed version of the APPL|F|T that does not penalize the violation of the total storage capacity of the distribution center. This solution strategy for the APPL|F|R, denoted with TEMPFIX, determines the zone temperature separately from the assignment of the quality change groups to the zones. A local search technique still changes the assignment of a temperature to a zone, but the assignment of the quality change groups to the zones is determined with a version of BFQUALFIX that does not check on the storage capacity constraints.

Solution strategy. In TEMPFIX, the neighborhood function of one of the local search techniques changes in each iterative improvement the temperature of one or more zones. Based on the assigned zone temperatures, the quality change groups are assigned to the zones with BFQUALFIX, assuming that each zone has ample storage capacity. The actual storage capacity $V_z'(z)$ of the individual zones is determined after the assignment of all the quality change groups and is equal to the maximum total assigned storage space, i.e., for each $z \in Z$

$$V_z'(z) = \max_{t \in T} \sum_{q \in Q | \hat{x}_z(q) = z} w_q(q, t),$$

(5.4)

where $w_q(q, t) \in \mathbb{N}_0^+$ is the group storage space requirement of quality change group $q \in Q$ in planning period $t \in T$, $\hat{x}_z(q) \in Z$ is the assigned zone for quality change group $q$. This equation is derived from (3.68). The adjusted zone volume $\zeta_z(z)$ for each zone $z \in Z$ is calculated by the application of (3.75). For the test instances this means that the zone volume is always three times the zone capacity. With the zone capacities and zone conditions known for each zone, we can now calculate the cost function of the APPL|F|R, which is the same as the cost function of the APPL with the penalty for exceeding the storage capacity $\pi_c$ equal to 0.
in (5.1). For TEMPFix, we used a random initial solution to start the local search. All initial solution obtained by randomly assigning temperatures to zones result in feasible solutions for the APPL|F|T with TEMPFix. The detailed implementation of the five local search techniques that are used for TEMPFix are described in Section 4.2.

**Algorithmic performance.** For the following experiments we used the 15 problem instances with three planning periods as described in Section 5.2. In Table 5.17, the results of the test for the APPL|F|R with TEMPFix in combination with the different local search techniques are given. The meaning of the columns is the same as in Table 5.6. Like in the previous experiments, we performed for each problem instance five runs using different seeds for the random number generator. Each run took a computation time of 120 seconds on a 200 MHz PentiumPro processor running under MS-DOS. We reduced the computation time for TEMPFix with 80% compared with the experiments with APPL|F|T since we only have to assign temperatures. The decision space with only temperatures is much smaller than in the case of the APPL|F|T. The BFQUALFix that assigns the quality change groups requires only a small amount of time before each evaluation of the cost function. Trail runs with 600 seconds did not find better solutions, which confirmed that 120 seconds was sufficient.

Table 5.17: Results of TEMPFix for the APPL|F|R with five different local search techniques for problem instances with three planning periods. The meaning of the other columns is the same as in Table 5.6.

<table>
<thead>
<tr>
<th></th>
<th>Repeated descent</th>
<th>Tabu search</th>
<th>Simulated annealing</th>
<th>Parallel descent</th>
<th>Genetic algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>min</td>
<td>avg</td>
<td>min</td>
<td>avg</td>
<td>min</td>
</tr>
<tr>
<td>Z2A3</td>
<td>838</td>
<td>838</td>
<td>838</td>
<td>838</td>
<td>838</td>
</tr>
<tr>
<td>Z2B3</td>
<td>838</td>
<td>838</td>
<td>838</td>
<td>838</td>
<td>838</td>
</tr>
<tr>
<td>Z2R3</td>
<td>838</td>
<td>838</td>
<td>838</td>
<td>838</td>
<td>838</td>
</tr>
<tr>
<td>Z3A3</td>
<td>639</td>
<td>641</td>
<td>639</td>
<td>645</td>
<td>639</td>
</tr>
<tr>
<td>Z3B3</td>
<td>641</td>
<td>642</td>
<td>639</td>
<td>642</td>
<td>639</td>
</tr>
<tr>
<td>Z3R3</td>
<td>639</td>
<td>642</td>
<td>642</td>
<td>645</td>
<td>639</td>
</tr>
<tr>
<td>Z4A3</td>
<td>489</td>
<td>496</td>
<td>502</td>
<td>508</td>
<td>485</td>
</tr>
<tr>
<td>Z4B3</td>
<td>490</td>
<td>497</td>
<td>500</td>
<td>506</td>
<td>485</td>
</tr>
<tr>
<td>Z4R3</td>
<td>490</td>
<td>496</td>
<td>489</td>
<td>509</td>
<td>485</td>
</tr>
<tr>
<td>Z6A3</td>
<td>417</td>
<td>421</td>
<td>427</td>
<td>432</td>
<td>413</td>
</tr>
<tr>
<td>Z6B3</td>
<td>421</td>
<td>431</td>
<td>426</td>
<td>432</td>
<td>413</td>
</tr>
<tr>
<td>Z6R3</td>
<td>413</td>
<td>417</td>
<td>421</td>
<td>423</td>
<td>402</td>
</tr>
<tr>
<td>Z8A3</td>
<td>388</td>
<td>394</td>
<td>376</td>
<td>381</td>
<td>367</td>
</tr>
<tr>
<td>Z8B3</td>
<td>387</td>
<td>396</td>
<td>381</td>
<td>390</td>
<td>367</td>
</tr>
<tr>
<td>Z8R3</td>
<td>359</td>
<td>365</td>
<td>346</td>
<td>355</td>
<td>343</td>
</tr>
</tbody>
</table>
Table 5.18: The derived optimized conditions in the zones by the use of TEMPFIX for the problem instances based on dataset R with three planning periods. The column ‘\(\hat{\mu}\)’ presents the minimal required total storage capacity and ‘\(\varepsilon(z)\)’ the maximum ethylene concentration in ppm during the year. The meaning of the other columns is the same as in Table 5.1.

| 2 | 1443 | 1 | 408 | 8.0 | 0.43 |
| 2 | 1035 | 0.0 | 0.00 |
| 3 | 1454 | 1 | 219 | 10.0 | 1.06 |
| 2 | 200 | 8.0 | 0.21 |
| 3 | 1035 | 0.0 | 0.00 |
| 4 | 1455 | 1 | 219 | 10.0 | 1.06 |
| 2 | 116 | 8.0 | 0.25 |
| 3 | 144 | 4.0 | 0.05 |
| 4 | 978 | 0.0 | 0.00 |
| 6 | 1474 | 1 | 172 | 11.1 | 1.50 |
| 2 | 64 | 10.0 | 0.69 |
| 3 | 96 | 8.0 | 0.20 |
| 4 | 20 | 7.0 | 0.45 |
| 5 | 144 | 4.0 | 0.05 |
| 6 | 978 | 0.0 | 0.00 |
| 8 | 1474 | 1 | 172 | 11.1 | 1.50 |
| 2 | 64 | 10.0 | 0.69 |
| 3 | 40 | 9.0 | 0.02 |
| 4 | 56 | 8.0 | 0.33 |
| 5 | 20 | 7.0 | 0.45 |
| 6 | 70 | 4.0 | 0.10 |
| 7 | 74 | 2.0 | 0.00 |
| 8 | 978 | 0.0 | 0.00 |

Discussion. Without the storage capacity constraints, TEMPFIX finds for the APPL|F|R lower cost for all the problem instances with three zones irrespective the local search technique that is used. On average the zone capacities of the derived solutions are 20% smaller, since the zone capacities are determined after the assignment of the quality change groups. Recall that the average sum of the total storage space requirement in a planning period was 80% of the original total storage capacity. Both the changes in storage capacity and conditions of the zones for the problem
instances based on dataset R are shown in Table 5.18. We remark that the calculated zone capacities for the zones with temperatures around 8° C are, in most cases, too small to be implemented in practice.

In Table 5.18 it can be observed that the most important change in temperature occurs in the solution for the problem instance with three zones. The temperature in the second zone is increased from 4° C to 8° C. This is roughly the same improvement that we found with the APPL|F|T for this problem instance. Regardless the number of available zones, the ethylene concentration remains in all solutions below the critical concentration of 1.5 ppm. Therefore, we conclude that the temperature effect on the keeping quality is the main driving force behind the decision to create more zones in a distribution center for perishables. In the case of assortments that are smaller than the one we studied here, it results only in fewer number of zones if there are not any products with a tropical or subtropical origin. Products such as bananas in the assortment justify at least three zones or a special treatment in the distribution center such as insulating blankets against the cold.

5.6 Operational assignment

In the previous sections we discussed models that are able to assign articles to slots in a static way. This kind of studies are performed on a tactical level. For the day-to-day operations we require a more dynamic assignment procedure. This in order to deal with the daily fluctuations in purchases and customer orders and the resulting varying stock levels in the distribution center. This problem has been studied with a simulation model that examines the effect of storage policies and a slot plan on the efficiency and effectiveness of a distribution center for perishables.

5.6.1 Simulation model of storage and retrieval

The influence of the daily requests for storage and retrieval during a whole year are studied with a simulation model that assigns purchases of quality change groups to zones. The daily putaways and retrievals follow the seasonal production and demand of the quality change groups. We concentrated for the storage and retrieval advice on zones instead of slots since we are in this section more interested in keeping quality and less in the consequences for handling. All requests are handled on a first come first served basis.
We examined the following three storage policies with the simulation model.

1. *Free zone policy.* The article is stored in the zone with the largest amount of unoccupied storage space.

2. *Temperature zone policy.* The article is stored in a zone where the storage temperature is closest to the optimal storage temperature of the quality change group.

3. *Preferred zone policy.* The preferred zone to store the article is determined by the slot plan. If the preferred zone has no available storage space left, we search for a zone with available storage space that is least harmful for the article. This advice assumes that the ethylene concentration is below 1.5 ppm.

All three storage policies are derived from the closest available slot policy. The first two are based on existing practices in distribution centers for perishables. When all zones are occupied, the purchase is stored in an overflow area with ambient temperature and enough ventilation. The free zone policy needs only the overflow area as alternative zone. This storage policy only focuses on storage capacity utilization. The list of the alternative zones in the temperature zone policy is determined by sorting the zones on the absolute deviation of the optimal storage temperature of the quality change group and the storage temperature in the zone. When the zone is full, we check the next zone in the list. The temperature zone policy requires only a little amount of additional information about the keeping quality characteristics to reduce the keeping quality loss. The slot plan for the last policy is found by the solution strategy introduced in this chapter. The sequence of alternative least harmful zones is found by sorting the zones on expected keeping quality loss $\Lambda(q, z, e(z))$, without checking on available storage capacity and the presence of ethylene. This storage policy is the most elaborate of all three examined storage policies.

In the case of a retrieval, we only follow the First In First Out rule when no alternative zones are occupied with the quality change group. Stock in alternative zones with harmful storage conditions for the quality change group has priority over stock in the assigned or default zone and is therefore used first for a retrieval.

In this section we are interested in the effect of a slot plan on the keeping quality loss and the use of alternative zones. Alternative zones require additional search efforts of the personnel or an accurate real-time locator system. We assume that the expected handling time for a handling request
is equal in the various zones. During the simulation run we calculate for each day the maximum relative keeping quality loss for all quality change groups that stay overnight in the distribution center with the following definition.

**Definition 5.3** (Maximum relative keeping quality loss). The maximum relative keeping quality loss $\lambda$ is calculated over all quality change groups $q \in Q$ and over all zones $z \in Z$ where articles of that particular quality change group are stored with respect to the ethylene concentration, i.e.,

$$\lambda = \omega_q \sum_{q \in Q} \max_{z \in Z} (\text{sign}(\tilde{w}_q(q, z)) \cdot \Lambda(q, z, \varepsilon(z))) \, ,$$

(5.5)

where $\tilde{w}_q(q, z) \in \mathbb{N}_0^+$ is the amount of a quality change group $q$ that is stored overnight in a zone $z$, $\varepsilon(z) \in \mathbb{R}^+$ is the ethylene concentration during the night in a zone, and $\omega_q \in \mathbb{N}^+$ is the weightfactor for keeping quality of a slot plan.

For each day the number of violations of the maximum allowed relative keeping quality loss $\Lambda_{max}$, as stated in (3.69), is calculated as a percentage of all quality change groups. This *violation percentage* is denoted with $\Lambda_{vm}$.

We measure the efficiency of the operations by the percentage of putaway requests that require an alternative zone instead of the assigned zone compared with the total number of putaway requests. This *alternative storage percentage* is denoted with $\eta_{az}$. Measured performance indicators for the effectiveness of the operations are the average of $\lambda$ during the whole year, denoted by $\overline{\lambda}$, and the average of $\Lambda_{vm}$ during the whole year, denoted by $\overline{\Lambda}_{vm}$.

### 5.6.2 Results of the simulation study

To test the three storage policies, we used the real world dataset from the wholesaler of vegetables and fruits described in Section 2.5. We simulated the operations of the wholesaler at the quality change group level with generated daily purchases and shipments based on the wholesaler dataset of a whole year. The optimal slot plans of APPL|F are found with tabu search, as presented in Tables 5.7 - 5.15.

Note that the system is never in a steady state since the distribution center of the wholesaler is about empty on sundays. Therefore, we assumed no problems with instability during the startup of each simulation run.
Table 5.19: The results of the simulation of storage and retrieval with three different storage policies. The cost of the optimal slot plan for a problem instance is presented in the 'Plan' column. The '$\eta_{az}$' column represents the alternative storage percentage. The '̅$\lambda$' column presents the average $\lambda$ as described in (5.5) and the '̅$\Lambda_{vm}$' column the average violation percentage.

<table>
<thead>
<tr>
<th>Plan</th>
<th>Storage policy</th>
<th>$\eta_{az}$</th>
<th>̅$\lambda$</th>
<th>̅$\Lambda_{vm}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Z2R3</td>
<td>838</td>
<td>0.0</td>
<td>4549</td>
<td>35.5</td>
</tr>
<tr>
<td></td>
<td>Free zone</td>
<td>0.5</td>
<td>969</td>
<td>3.5</td>
</tr>
<tr>
<td></td>
<td>Temperature zone</td>
<td>0.4</td>
<td>757</td>
<td>0.4</td>
</tr>
<tr>
<td>Z3R3</td>
<td>698</td>
<td>0.0</td>
<td>4439</td>
<td>42.3</td>
</tr>
<tr>
<td></td>
<td>Free zone</td>
<td>0.0</td>
<td>1064</td>
<td>4.6</td>
</tr>
<tr>
<td></td>
<td>Temperature zone</td>
<td>0.1</td>
<td>564</td>
<td>0.0</td>
</tr>
<tr>
<td>Z4R3</td>
<td>485</td>
<td>0.0</td>
<td>4620</td>
<td>46.7</td>
</tr>
<tr>
<td></td>
<td>Free zone</td>
<td>0.2</td>
<td>706</td>
<td>2.0</td>
</tr>
<tr>
<td></td>
<td>Temperature zone</td>
<td>0.0</td>
<td>403</td>
<td>0.1</td>
</tr>
<tr>
<td>Z6R3</td>
<td>413</td>
<td>0.0</td>
<td>4352</td>
<td>40.2</td>
</tr>
<tr>
<td></td>
<td>Free zone</td>
<td>2.1</td>
<td>739</td>
<td>3.6</td>
</tr>
<tr>
<td></td>
<td>Temperature zone</td>
<td>0.2</td>
<td>344</td>
<td>0.0</td>
</tr>
<tr>
<td>Z8R3</td>
<td>367</td>
<td>0.0</td>
<td>4343</td>
<td>40.9</td>
</tr>
<tr>
<td></td>
<td>Free zone</td>
<td>6.3</td>
<td>939</td>
<td>6.7</td>
</tr>
<tr>
<td></td>
<td>Temperature zone</td>
<td>0.4</td>
<td>323</td>
<td>0.1</td>
</tr>
</tbody>
</table>

The results of the simulation are shown in Table 5.19. The first value of each policy gives the alternative storage percentage. The maximum relative keeping quality loss $\lambda$ as described in (5.5) is presented as the average over the year in the ̅$\lambda$ column. The violation percentage is presented as '̅$\Lambda_{vm}$' in the last column. All three values are the average over five simulation runs of a whole year.

The simulation results in Table 5.19 show for all problem instances that the preferred zone policy has the best performance compared to the other two policies. The value of ̅$\lambda$ is lower than the optimal value of the static slot plan in the preferred zone policy since not all quality change groups are present in the assortment at the same time. Ignoring the need for specific storage conditions in the free zone policy results in too much unnecessary keeping quality loss in the distribution center. The use of a slot plan compared to a simple quality oriented assignment rule in the temperature zone policy reduces the maximum keeping quality loss with 46 %. The
violation percentage is sharply reduced. The temperature zone policy has the highest alternative storage percentage of all three storage policies.

5.7 Conclusions

We proved by implementation of our concept of a slot plan for perishables that such a concept is feasible in practice. Computational studies showed that our solution strategy results in feasible and often optimal solutions.

The number of zones that are needed in the distribution center depends mainly on the allowed level of relative keeping quality loss and the effect of temperature and ethylene on the quality change groups in the assortment. For most vegetables and fruits assortments, we recommend at least three zones, but we prefer four zones. When we can redesign the layout, the procedure TEMPFIX finds slot plans with a cost lower than in the case of a fixed layout with predefined storage conditions. The exact storage capacities and storage conditions found with TEMPFIX cannot always be implemented in practice.

Applying a slot plan for the operational assignment of quality change groups to zones seems useful as part of a storage policy in a distribution center for perishables. The additional effort to make a slot plan and integrating it into the storage policy is relatively small compared with the potential benefits from reduced keeping quality loss and less use of alternative zones. We recommend therefore the proposed ‘preferred zone policy’ for use in distribution centers for perishables. The assignment rules used in the proposed ‘preferred zone storage policy’ are not enough to ensure suitable storage conditions for all the articles, since personnel must follow the guidelines for storage and retrieval to avoid that articles get lost and suffer from unnecessary keeping quality loss. A real-time locator system is in theory able to reduce the number of additional searches in the distribution center associated with the use of alternative zones. In practice, the uniform article coding and barcodes needed for such a locator system are still difficult to implement for agricultural perishables.
Assignment of perishables to zones
In this chapter we concentrate on solution strategies for allocating storage space for articles to the forward pick storage. We handle the knapsack version of the forward-reserve problem (FRP) with local search techniques. After explaining the proposed construction algorithm and neighborhood function of the solution strategy, we present computational studies of several real-world case examples.

6.1 Introduction

In the previous chapter, we considered the assignment of quality change groups to zones. In this chapter, we focus on the allocation of storage space for the articles between the forward pick storage and the reserve storage in each zone. In this way, we obey the assignment of the articles in a quality change group to the zones, which we discussed in Chapter 5. In Section 3.1.4, we reformulated the FRP as a knapsack problem denoted by knapsack forward-reserve problem (KFRP), which closely resembles the multiply-choice knapsack problem as reviewed by Martello & Toth [1991]. For the KFRP, we have to determine for each article a set of possible storage allocations in the forward pick storage or slot sizes with the corresponding savings in handling time. The proposed solution strategy for
the KFRP is able to select at most one slot size for each article or exactly one slot size when we include zero slot sizes with zero savings in each set.

**Slot sizes.** The possible values for the slot sizes depend on the sizes of the packaging units and on the maximum size of a unit load volume. We assume for all articles the same type of standardized packaging units and therefore similar article sizes. Currently used packaging units such as returnable transport containers for vegetables are available with the following dimensions of the footprint: (0.6 m × 0.4 m), (0.4 m × 0.3 m), and (0.3 m × 0.2 m). These standardized modular packaging units are designed to fit on standard pallets with the dimensions (1.0 m × 1.2 m) or (0.8 m × 1.2 m). The type of storage equipment determines the maximum size of a unit load. A pallet rack with a slot for a maximum pallet size can be divided by a shelf in two slots with the storage capacity of half a maximum pallet size. Based on these packaging characteristics and storage equipment properties, we propose the following two types of sets of possible slot sizes.

- **Linear slot sizes.** All possible slots in the set have a size that is a multiple of a base size, usually the size of the smallest standard packaging unit.

- **Divisible slot sizes.** All possible slots in the set have a size that fits in a sequence of distinct sizes $s_1 < s_2 < \ldots < s_n$, where all the sizes are such that $s_i|s_{i+1}$ holds.

We expect that the standardization of the articles sizes will make the stacking of the articles on a unit load easier and possibly achieves a higher utilization of the forward pick storage.

**Average handling times.** The implementation of the FRP in practice requires that we elaborate the handling models introduced in Section 3.1.1 in such a way that we are able to determine the average handling times from the chosen layout and the storage capacity of the forward pick storage.

The savings in handling time mainly depend on the average travel time to the reserve storage, this relative to the average travel time for replenishment and forward picking. For the FRP, we made the assumption of a single location for the reserve slots and a single location for the forward pick slots. The distances from the i/o point to these locations correspond to the average travel distance from the i/o point into these warehousing systems, assuming uniform demand over the warehousing system. The
6.1. Introduction

travel distance for replenishment depends only on the perimeter of the warehouse. A replenishment operation moves between reserve storage and the forward pick storage, without necessarily visiting the i/o point. The other travel distances depend on both the chosen layout of the warehouse and the size of the forward pick storage with respect to the size of the total storage area. Moreover, when we also assume that all stock is located at the ground level, the length of the warehouse $D \in \mathbb{N}^+$, is directly related to the total storage capacity $\hat{V} \in \mathbb{N}^+$. The size of the forward pick storage is related to the forward pick storage capacity $\hat{V}_f \in \mathbb{N}^+$.

![Linear layout](image)

![Square layout](image)

Figure 6.1: Two schematic layouts for the storage area in a warehouse. The length $D$ is a measure for the size of the warehouse.

We considered the following two types of layout in our study of the FRP, which are illustrated in Figure 6.1.

- **Linear layout.** The locations in this layout are arranged in a single one sided aisle or along a fixed pick route. The length $D$ is assumed to be equal to the total storage capacity $\hat{V}$.

- **Square layout.** In this layout, the length of the sides of the warehouse are equal. The length of a side is assumed to be equal to the square root of the total storage capacity, i.e., $D/2 = \sqrt{\hat{V}}$. We assume in
this layout that the slots are located along aisles, which results in rectilinear travel distances between the locations.

The assumptions about the two warehousing systems lead to the following definitions for the average travel distances in the warehouse, based on the models of Christofides & Eilon [1969].

**Definition 6.1** (Average travel distance). The average travel distance depends on the warehousing system where the slot is located, on the storage capacity of the forward pick area $\hat{V}_f \in \mathbb{N}^+$ relative to the total storage capacity of the warehouse $\hat{V} \in \mathbb{N}^+$, and on the length of the warehouse $D \in \mathbb{N}^+$ according to the following expressions.

- The average travel distance from the i/o point to a forward pick slot $d_f$ is given by

\[
d_f = \frac{\hat{V}_f}{\hat{V}} \cdot \frac{D}{2}.
\]  

(6.1)

- The average travel distance from the i/o point to a reserve pick slot $d_r$ is given by

\[
d_r = \left(1 + \frac{\hat{V}_f}{\hat{V}}\right) \cdot \frac{D}{2}.
\]  

(6.2)

- The average travel distance between a slot in the reserve area and a slot in the forward pick area $d_s$ is given by

\[
d_s = \frac{D}{2}.
\]  

(6.3)

Recall that for replenishment, the picker does not visit the i/o point but shuttles between the reserve storage and the forward pick storage. Therefore, we obtain a travel distance for replenishment that is shorter than the travel distance for reserve picking.

To determine the handling time from the average travel distance, we need to define the fixed time needed for each pick cycle and the speed of the picker, depending on the type of equipment.

**Definition 6.2** (Fixed cycle time). The fixed cycle time $\theta_{\text{fix}} \in \mathbb{N}^+$ is given by the handling times for stopping at the slot plus the loading/unloading for each pick cycle.

**Definition 6.3** (Travel speed). The travel speed depends on the type of handling equipment, i.e.,
6.2. Problem generation

- $M_f \in \mathbb{N}^+$, the travel speed for a forward picker, and
- $M_r \in \mathbb{N}^+$, the travel speed for a reserve picker.

In the case that all variables in the Definitions 6.1-6.3 are known, we are able to calculate the average cycle times and the replenishment time according to

\[
\begin{align*}
\bar{\theta}_{\text{eff}} &= \theta_{\text{fix}} + \frac{2 \cdot d_f}{M_f}, \\
\bar{\theta}_{\text{cfr}} &= \theta_{\text{fix}} + \frac{2 \cdot d_f}{M_r}, \\
\bar{\theta}_{\text{cerr}} &= \theta_{\text{fix}} + \frac{2 \cdot d_r}{M_r}, \text{ and} \\
\bar{\theta}_{\text{rpl}} &= \theta_{\text{fix}} + \frac{2 \cdot d_s}{M_r},
\end{align*}
\]

where $\bar{\theta}_{\text{eff}} \in \mathbb{R}^+$ gives the average cycle time to a forward pick slot for a forward picker, $\bar{\theta}_{\text{cfr}} \in \mathbb{R}^+$ the average cycle time to a forward pick slot for a reserve picker, $\bar{\theta}_{\text{cerr}} \in \mathbb{R}^+$ the average cycle time to a reserve slot for a reserve picker, and $\bar{\theta}_{\text{rpl}} \in \mathbb{R}^+$ the average replenishment time. Recall that the replenishment time depends on the speed of the reserve picker's equipment.

6.2 Problem generation

In this study we used real world data of customer orders of four distribution centers for perishables in the Netherlands.

The first dataset of the distribution center of the wholesaler of vegetables and fruits, as described in Section 2.5, has important seasonal fluctuations. We have split this dataset for the KFRP in three datasets for each of the three seasons, denoted with W1, W2, and W3. Each dataset corresponds with a planning period in the distribution center with a different assortment for each planning period. We adopted the current policy of the wholesaler of a single forward pick storage in the main zone instead of separate forward pick storage areas or no forward pick storage in the different cold stores. Keeping quality loss was minimized by putting the stock back in the appropriate reserve storage at the end of each day, at least for the most vulnerable quality change groups.
The other three distribution centers belong to three different manufacturers of refrigerated food, such as cheese, sausage, snack salads and desserts. From these manufacturers we received datasets that represent all customer orders during a selected week. Analysis of the turnover during the whole year revealed no significant seasonal fluctuations in the turnover compared to the turnover during the selected week. The average storage time for all four datasets was 3.5 days. This corresponds with a yearly turnover rate of 104. Data on the total storage capacity and the service level of the distribution center was not available. The articles in the various assortments did not have important product interactions and therefore could be stored in a single zone at a temperature which is a compromise between the different quality change groups. We designated the datasets of the three manufacturers with M1, M2 and M3.

From these four original datasets, we constructed 72 problem instances by varying the type of slot sizes, the forward pick storage capacity $\hat{V}_I$, and the layout of the warehouse.

The slot sizes were based on the maximum pallet volume $\omega_u = 8$ and a minimum storage capacity of a slot equal to $1 = \omega_u/8$. This results in two basic sets of slot sizes given by

\[
\mathcal{W}_{\text{lin}} = \{0, 1, 2, 3, 4, 5, 6, 7, 8\} , \quad \text{and} \\
\mathcal{W}_{\text{div}} = \{0, 1, 2, 4, 8\} .
\] (6.8)\hspace{1cm} (6.9)

The size "0" in each set makes it possible to allocate zero storage space for an article. Note that the set $\mathcal{W}_{\text{lin}}$ is nearly twice as large as the set $\mathcal{W}_{\text{div}}$.

We created problem instances with three levels of forward pick storage capacity: A, B, and C. The chosen forward pick storage capacity was always less than the total storage capacity. The total storage capacity $\hat{V} \in \mathbb{N}^+$ was based on the total required storage capacity during planning period $t$ and an average utilization of 80 %, i.e.

\[
\hat{V} = \frac{1}{0.8} \cdot \sum_{a \in \mathcal{A}} w(a, t) .
\] (6.10)

The generated problem instances are based on a linear layout or on a square layout. The type of layout and the total storage capacity determine the dimensions of the warehouse. The two types of layout are denoted in the name of the instance with 'L' in the case of a linear layout and 'S' in the case of a square layout.
6.2. Problem generation

We used the following estimates for the parameters in the handling model described in Section 3.1.1 and elaborated in (6.1) - (6.7). The fixed cycle time $\theta_{\text{fix}} = 60$ s. The travel speed of the forward picker $M_f = 0.25$ m/s, and the travel speed of the reserve picker $M_r = 1$ m/s. The extract time for a unit load of a reserve picker $\theta_{\text{gar}} = 3$ s, the grab time for an article unit of a forward picker $\theta_{\text{gaf}} = 3$ s, and the grab time for an article unit of a reserve picker $\theta_{\text{gar}} = 10$ s.

The name of the problem instance contains information about the way the instance was generated. For instance, M2BS denotes the problem instance based on the dataset of manufacturer M2 with the forward pick storage capacity level B, and a square layout. The characteristics of the problem instances that are independent of the set of slot sizes are presented in Table 6.1 for the instances with a linear layout and Table 6.2 for the instances with a square layout.

Table 6.1: Characteristics of the problem instances with a linear layout.

<table>
<thead>
<tr>
<th></th>
<th>$\hat{V}$</th>
<th>$\hat{V}_f$</th>
<th>$A$</th>
<th>$\theta_{\text{cr}}$</th>
<th>$\theta_{\text{crf}}$</th>
<th>$\theta_{\text{cfr}}$</th>
<th>$\theta_{\text{eff}}$</th>
<th>$\theta_{\text{rpl}}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>W1AL</td>
<td>11039</td>
<td>464</td>
<td>1591</td>
<td>1857</td>
<td>133</td>
<td>350</td>
<td>1785</td>
<td></td>
</tr>
<tr>
<td>W1BL</td>
<td>928</td>
<td>1590</td>
<td></td>
<td>1930</td>
<td>205</td>
<td>640</td>
<td>1785</td>
<td></td>
</tr>
<tr>
<td>W1CL</td>
<td>1856</td>
<td>1547</td>
<td></td>
<td>2075</td>
<td>350</td>
<td>1220</td>
<td>1785</td>
<td></td>
</tr>
<tr>
<td>W2AL</td>
<td>10606</td>
<td>472</td>
<td>1598</td>
<td>1791</td>
<td>134</td>
<td>355</td>
<td>1717</td>
<td></td>
</tr>
<tr>
<td>W2BL</td>
<td>944</td>
<td>1598</td>
<td></td>
<td>1865</td>
<td>208</td>
<td>650</td>
<td>1717</td>
<td></td>
</tr>
<tr>
<td>W2CL</td>
<td>1888</td>
<td>1549</td>
<td></td>
<td>2012</td>
<td>355</td>
<td>1240</td>
<td>1717</td>
<td></td>
</tr>
<tr>
<td>W3AL</td>
<td>10122</td>
<td>456</td>
<td>1478</td>
<td>1713</td>
<td>131</td>
<td>345</td>
<td>1642</td>
<td></td>
</tr>
<tr>
<td>W3BL</td>
<td>912</td>
<td>1478</td>
<td></td>
<td>1784</td>
<td>203</td>
<td>630</td>
<td>1642</td>
<td></td>
</tr>
<tr>
<td>W3CL</td>
<td>1824</td>
<td>1441</td>
<td></td>
<td>1927</td>
<td>345</td>
<td>1200</td>
<td>1642</td>
<td></td>
</tr>
<tr>
<td>M1AL</td>
<td>3875</td>
<td>104</td>
<td>260</td>
<td>682</td>
<td>76</td>
<td>125</td>
<td>665</td>
<td></td>
</tr>
<tr>
<td>M1BL</td>
<td>208</td>
<td>260</td>
<td></td>
<td>698</td>
<td>93</td>
<td>190</td>
<td>665</td>
<td></td>
</tr>
<tr>
<td>M1CL</td>
<td>416</td>
<td>256</td>
<td></td>
<td>730</td>
<td>125</td>
<td>320</td>
<td>665</td>
<td></td>
</tr>
<tr>
<td>M2AL</td>
<td>5912</td>
<td>152</td>
<td>240</td>
<td>1007</td>
<td>84</td>
<td>155</td>
<td>984</td>
<td></td>
</tr>
<tr>
<td>M2BL</td>
<td>304</td>
<td>240</td>
<td></td>
<td>1031</td>
<td>108</td>
<td>250</td>
<td>984</td>
<td></td>
</tr>
<tr>
<td>M2CL</td>
<td>608</td>
<td>239</td>
<td></td>
<td>1079</td>
<td>155</td>
<td>440</td>
<td>984</td>
<td></td>
</tr>
<tr>
<td>M3AL</td>
<td>4974</td>
<td>248</td>
<td>815</td>
<td>876</td>
<td>99</td>
<td>215</td>
<td>837</td>
<td></td>
</tr>
<tr>
<td>M3BL</td>
<td>496</td>
<td>810</td>
<td></td>
<td>915</td>
<td>138</td>
<td>370</td>
<td>837</td>
<td></td>
</tr>
<tr>
<td>M3CL</td>
<td>992</td>
<td>802</td>
<td></td>
<td>992</td>
<td>215</td>
<td>680</td>
<td>837</td>
<td></td>
</tr>
</tbody>
</table>

It can be observed in Tables 6.1 and 6.2 that the cycle times increase with
Table 6.2: Characteristics of the problem instances with a square layout.

<table>
<thead>
<tr>
<th></th>
<th>$\hat{V}$</th>
<th>$\hat{V}_f$</th>
<th>$A$</th>
<th>Cycle times</th>
<th>$\theta_{crr}$</th>
<th>$\theta_{cfr}$</th>
<th>$\theta_{cff}$</th>
<th>$\theta_{rpl}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>W1AS</td>
<td>11039</td>
<td>464 986</td>
<td></td>
<td>147 63 74 143</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>W1BS</td>
<td>928 952</td>
<td></td>
<td></td>
<td>150 67 88 143</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>W1CS</td>
<td>1856 860</td>
<td></td>
<td></td>
<td>157 74 116 143</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>W2AS</td>
<td>10606</td>
<td>472 902</td>
<td></td>
<td>145 64 74 141</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>W2BS</td>
<td>944 868</td>
<td></td>
<td></td>
<td>149 67 89 141</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>W2CS</td>
<td>1888 776</td>
<td></td>
<td></td>
<td>156 74 118 141</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>W3AS</td>
<td>10122</td>
<td>456 906</td>
<td></td>
<td>143 64 74 140</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>W3BS</td>
<td>912 866</td>
<td></td>
<td></td>
<td>147 67 89 140</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>W3CS</td>
<td>1824 792</td>
<td></td>
<td></td>
<td>154 74 117 140</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>M1AS</td>
<td>3875</td>
<td>104 187</td>
<td></td>
<td>111 61 65 109</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>M1BS</td>
<td>208 184</td>
<td></td>
<td></td>
<td>112 63 71 109</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>M1CS</td>
<td>416 167</td>
<td></td>
<td></td>
<td>114 65 81 109</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>M2AS</td>
<td>5912</td>
<td>152 236</td>
<td></td>
<td>122 62 66 121</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>M2BS</td>
<td>304 236</td>
<td></td>
<td></td>
<td>124 63 73 121</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>M2CS</td>
<td>608 236</td>
<td></td>
<td></td>
<td>127 66 85 121</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>M3AS</td>
<td>4974</td>
<td>248 673</td>
<td></td>
<td>119 63 71 116</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>M3BS</td>
<td>496 646</td>
<td></td>
<td></td>
<td>121 66 82 116</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>M3CS</td>
<td>992 597</td>
<td></td>
<td></td>
<td>127 71 104 116</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The size of the forward pick storage but that the replenishment time $\bar{\theta}_{rpl}$ remains the same. The pick times do not increase as much with the size of the forward pick storage as in the case of a linear layout. The cycle times $\theta_{crr}$, $\theta_{cfr}$, and $\theta_{cff}$ increase only slightly with respect to the size of the forward pick storage in a square layout compared to a linear layout. The replenishment time is independent of the size of the forward pick storage. The ratio

$$R = \frac{\beta \cdot \theta_{eff} + \bar{\theta}_{rpl}}{\beta \cdot \theta_{crr}}$$

calculates the relation between the handling times needed for the 'reserve and forward pick storage' situation and the handling time for the 'reserve storage only' situation. The ratio is based on the assumption that each handling request occurs with the same frequency and that we have to replenish after a number of forward pick requests equal to the unit load quantity $\beta$. For a given unit load quantity, a ratio greater than one
promises savings when we allocate storage space for that article to the reserve and the forward pick storage compared to only allocating storage space to the reserve storage.

![Graph showing the ratio R with β = 2 on the size of the forward pick storage in a linear layout and in a square layout for the dataset M3.](image)

**Figure 6.2:** The dependence of the ratio $R$ with $\beta = 2$ on the size of the forward pick storage in a linear layout and in a square layout for the dataset M3.

Figure 6.2 shows the ratio with $\beta$ equal to 2 for increasing sizes of the forward pick storage in the two types of layout. The linear layout promises higher savings from a forward pick storage in a linear layout than in a square layout. The difference diminishes when we increase the forward pick storage capacity relative to the total storage capacity. When the replenishment pickers have to visit the i/o point, then we have a large increase in replenishment time and a decrease in the ratio $R$. We still have a ratio greater than 1 for articles with greater unit load quantities.

Note that we assumed single-address forward picking in the handling model. We expect that multi-address forward picking instead of single address forward picking has a large effect on savings since this reduces the cycle time for a forward picker $\theta_{\text{eff}}$ considerably.
6.3 Solution strategy for the KFRP

Since we included for each article possible slots with zero slot sizes in all problem instances described in Section 6.2, we can always find a feasible solution for the KFRP by assigning these slots to the forward pick storage. We propose a greedy construction algorithm for the KFRP that results in reasonable good solutions. We expect that the initial solutions constructed with this greedy algorithm can be improved by local search.

6.3.1 Construction of initial solutions

We propose the Greedy Elimination (GREL) algorithm for the construction of initial solutions for the local search. Since the KFRP resembles the knapsack problem, we developed GREL as an extension of the popular greedy algorithm for the knapsack problem, described among others by Martello & Toth [1991]. Moreover, GREL is similar to the Greedy Knapsack Heuristic proposed by Van den Berg [1996] for the knapsack version of the FRP.

---

**Greedy Elimination algorithm (GREL)**

1. Let $A := |A|$, $u \in U = \bigcup_{a \in A} G_{a,t}$, and $N := |U| = \sum_{a \in A} |G_{a,t}|$. Sort the slots $u_1, \ldots, u_N$, such that

   $$\Theta_{\text{save}}(a, t, u_{\phi(1)})/v_f(u_{\phi(1)}) \geq \cdots \geq \Theta_{\text{save}}(a, t, u_{\phi(N)})/v_f(u_{\phi(N)}) ,$$

   with $\phi : \mathbb{N} \to \mathbb{N}$. Let $\bar{V} := \tilde{v}_f$, and let for all $u \in U$ $x_u(u) := 0$. Start with the slot with the highest saving per unit storage space by setting $i := 1$.

2. If $\bar{V} - v_f(u_{\phi(i)}) \geq 0$, then assign the slot to the forward pick storage: $x_u(u_{\phi(i)}) := 1$, and $\bar{V} := \bar{V} - v_f(u_{\phi(i)})$.

3. Remove the other slots of the assigned article from the remainder of the slot list $U$: for all $u_{\phi(j)} \in G_{a,t}$ and $j > i$ do

   $$U := U \setminus \{u_{\phi(j)}\} \text{ and } N := N - 1 .$$

4. Let $i := i + 1$. If $i \leq N$, then go to 2. Otherwise stop.
In Step 1, we sort the possible slots on the expected savings per unit of storage space. The highest savings relative to the associated storage space requirement get the highest priority for the assignment to the forward pick storage. Only possible slots that fit in the remaining storage capacity $\bar{V}$ are assigned. By removing the remaining slots of an article after an assignment in Step 3, we should be able to find solutions for the KFRP with exactly one slot or storage allocation for each article. The time complexity of GREL is estimated by the following theorem.

**Theorem 6.1.** GREL finds in $O(N \cdot \log N)$ time a solution for the KFRP.

**Proof.** Step 1 of the above outlined GREL requires $O(N)$ time for the determination of both the parameters $\Theta_{\text{save}}(a, t, u)$ and $v_f(u)$ and the initialization of the assignment $x_u(u)$. Sorting the variables takes $O(N \cdot \log N)$ time. Assigning a possible slot as described by Step 2 is performed at most $A$ times. The assignment can be carried out in constant time, however the removing of the remaining possible slots of an article in Step 3 can take at most $O(\max_{a \in A} |G_{a,t}|)$ time. Since the number of possible slots $N = \sum_{a \in A} |G_{a,t}|$ is less than or equal to $A \cdot \max_{a \in A} |G_{a,t}|$, Steps 2 and 3 require therefore $O(N)$ time. Hence, the time complexity of GREL is $O(N \cdot \log N)$.

Hence, GREL has the same time complexity as the original greedy algorithm for the knapsack problem with $N$ items.

**6.3.2 Neighborhood function**

The neighborhood function of a local search algorithm consists of a set of moves one exchange and swap that change the current solution into a set of neighbors or candidate solutions. We cannot guarantee feasible solutions for the KFRP when we assign for each article randomly a possible slot to the forward pick storage. A greedy algorithm such as GREL guarantees feasible solutions for the KFRP. Therefore, we developed a hybrid neighborhood function that uses a modification of GREL and the original moves in the neighborhood functions of the local search techniques tabu search and parallel descent, described in Section 4.2. Instead of directly assigning possible slots for each article to the forward pick storage, we assign a priority value to each possible slot. These priority values are applied for the sorting of the possible slots in Step 1 of GREL instead of the expected savings per unit of storage space. The moves in
the neighborhood function of the local search techniques change or swap priorities. Repeated application of the moves from tabu search or parallel descent makes it possible to visit all feasible solutions in the search space. Hence, in an indirect way, the assignment of the articles to the forward pick storage is carried out by using a construction algorithm.

6.3.3 Upper bound for the KFRP

We can derive a upper bound for the KFRP when we relax the integer assignments. The continuous relaxation of KFRP, denoted by $\mathcal{C}(\text{KFRP})$, is defined as follows.

**Definition 6.4 (Continuous relaxation of KFRP).** Given is an instance of the continuous relaxation of the KFRP by a six-tuple $KD = (\mathcal{A}, \mathcal{T}, \hat{V}_t, \mathcal{G}, v, \Theta_{\text{save}})$, as described in Definition 3.17. We define a knapsack assignment $\bar{x}_u$, where

$$\bar{x}_u : \mathcal{G} \rightarrow [0,1] \subset \mathbb{R} \quad (6.11)$$

assigns an article in a planning period to a forward pick slot. The problem is to find a knapsack assignment $\bar{x}_u$ that maximizes the total savings in (3.39) and satisfies (3.40)-(3.41).

According to Martello & Toth [1991], the cost of $\mathcal{C}(\text{KFRP})$ serves as an upper bound for instances of KFRP because we can satisfy the capacity constraint at equality. When we have for each article possible slots with zero storage space and zero savings, then we can find the cost of $\mathcal{C}(\text{KFRP})$ by assigning a fractional value to the first variable in the GREL for which the assignment would violate the capacity constraint.

6.3.4 Reduction of the problem size

According to Sinha & Zoltners [1979], we can reduce the number of slots in a set for an article in the problem instance of the KFRP, as described in Definition 3.17, by applying the following two dominance criteria.

**Dominance Criterion 6.1.** Given is the set of possible forward pick slots $\mathcal{G}_{a,t} \subset \mathcal{F}$, the handling time function $\Theta_{\text{save}}(a,t,s) \in \mathbb{N}$ and the slot size function $v(s) \in \mathbb{N}^+$ for each article $a \in \mathcal{A}$, planning period $t \in \mathcal{T}$, and slot $s \in \mathcal{G}_{a,t}$. For any possible set of forward pick slots $\mathcal{G}_{a,t}$, if there exist two slots $i, j \in \mathcal{G}_{a,t}$ such that $v(i) \geq v(j)$ and $\Theta_{\text{save}}(a,t,i) \leq \Theta_{\text{save}}(a,t,j)$, then there exists an optimal solution for KFRP in which $x_a(i) = 0$, i.e., slot $i$ is dominated. □
6.4. Computational results

A further reduction of the size of the problem instances of C(KFRP) is achieved by applying the next criterion.

**Dominance Criterion 6.2.** Given is the set of possible forward pick slots \( G_{a,t} \subset \mathcal{F} \), the handling time function \( \Theta_{\text{save}}(a,t,s) \in \mathbb{N} \) and the slot size function \( v(s) \in \mathbb{N}^+ \) for each article \( a \in A \), planning period \( t \in T \), and slot \( s \in G_{a,t} \). For any \( G_{a,t} \), if there exist three slots \( h, i, j \in G_{a,t} \), such that \( v(h) < v(i) < v(j) \) and

\[
\frac{\Theta_{\text{save}}(a,t,i) - \Theta_{\text{save}}(a,t,h)}{v(i) - v(h)} \leq \frac{\Theta_{\text{save}}(a,t,j) - \Theta_{\text{save}}(a,t,i)}{v(j) - v(i)},
\]

then there exists an optimal solution to C(KFRP) in which \( \tilde{x}_u(i) = 0 \), i.e., slot \( i \) is dominated. \( \square \)

Proofs of both criteria can be found in Sinha & Zoltners [1979]. We applied both criteria during the generation of the problem instances by sorting first the slots of each article according to increasing sizes.

6.4 Computational results of the KFRP

We employ computational studies to examine the effects of the forward pick storage capacity relative to the total storage capacity, the type of layout and the type of slot sizes on the maximum savings in handling time in a warehouse.

**Algorithmic performance.** Tables 6.3 - 6.6 present the test results for each set of problem instances with tabu search and parallel descent. We used the implementation of tabu search and parallel descent described in Section 4.2. In the column 'set', with 'L' is denoted the application of the set of linear slot sizes and with 'D' is denoted the application of the set of divisible slot sizes. For each problem instance, 'N' denotes the number of possible slots and 'UBLP' the optimal cost of C(KFRP) for the problem instance, i.e., the upper bound. The value of UBLP was calculated with the simplex method in the solver Xpress-MP of Dash Associates. For each problem instance we performed five runs using different seeds for the random number generator. Each run of a local search technique started with an initial solution constructed by the GREL algorithm, as stated by the column 'init'. The column 'max' presents the maximal cost and 'avg' presents the average cost of five runs. Solutions with a cost equal to the integer value of the upper bound are considered optimal. All cost are expressed by seconds per day. Each run took a computation time of
Table 6.3: Results of the KFRP with tabu search and parallel descent for problem instances of the wholesaler. In the ‘set’ column, ‘L’ indicates the use of the set of linear slot sizes and ‘D’ the set of divisible slot sizes. The column ‘N’ gives the total number of possible slots, the column ‘UB_{LP}’ the optimal cost of C(KFRP) and the ‘init’ column the initial solution constructed with the GREL algorithm. The column ‘max’ gives the maximal cost and ‘avg’ the average cost over five runs. All cost are given in seconds per day.

<table>
<thead>
<tr>
<th>set</th>
<th>N</th>
<th>UB_{LP}</th>
<th>init</th>
<th>Tabu search</th>
<th>Parallel descent</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>max</td>
<td>avg</td>
</tr>
<tr>
<td>W1AL</td>
<td>L</td>
<td>4087</td>
<td>14555.0</td>
<td>14010</td>
<td>14256</td>
</tr>
<tr>
<td>W1AL</td>
<td>D</td>
<td>2633</td>
<td>14453.5</td>
<td>13928</td>
<td>14299</td>
</tr>
<tr>
<td>W1AS</td>
<td>L</td>
<td>2227</td>
<td>1211.0</td>
<td>1136</td>
<td>1195</td>
</tr>
<tr>
<td>W1AS</td>
<td>D</td>
<td>1863</td>
<td>1200.5</td>
<td>1083</td>
<td>1176</td>
</tr>
<tr>
<td>W1BL</td>
<td>L</td>
<td>3933</td>
<td>17630.0</td>
<td>16092</td>
<td>16907</td>
</tr>
<tr>
<td>W1BL</td>
<td>D</td>
<td>2532</td>
<td>17338.0</td>
<td>16015</td>
<td>17062</td>
</tr>
<tr>
<td>W1BS</td>
<td>L</td>
<td>2133</td>
<td>1618.0</td>
<td>1478</td>
<td>1588</td>
</tr>
<tr>
<td>W1BS</td>
<td>D</td>
<td>1783</td>
<td>1597.0</td>
<td>1459</td>
<td>1567</td>
</tr>
<tr>
<td>W1CL</td>
<td>L</td>
<td>3360</td>
<td>16053.0</td>
<td>14228</td>
<td>15226</td>
</tr>
<tr>
<td>W1CL</td>
<td>D</td>
<td>2161</td>
<td>15477.0</td>
<td>13703</td>
<td>15176</td>
</tr>
<tr>
<td>W1CS</td>
<td>L</td>
<td>1864</td>
<td>1913.0</td>
<td>1571</td>
<td>1813</td>
</tr>
<tr>
<td>W1CS</td>
<td>D</td>
<td>1507</td>
<td>1819.0</td>
<td>1385</td>
<td>1768</td>
</tr>
<tr>
<td>W2AL</td>
<td>L</td>
<td>3726</td>
<td>12766.6</td>
<td>12180</td>
<td>12512</td>
</tr>
<tr>
<td>W2AL</td>
<td>D</td>
<td>2514</td>
<td>12674.5</td>
<td>12155</td>
<td>12551</td>
</tr>
<tr>
<td>W2AS</td>
<td>L</td>
<td>2041</td>
<td>1340.3</td>
<td>1180</td>
<td>1291</td>
</tr>
<tr>
<td>W2AS</td>
<td>D</td>
<td>1676</td>
<td>1308.5</td>
<td>1126</td>
<td>1285</td>
</tr>
<tr>
<td>W2BL</td>
<td>L</td>
<td>3570</td>
<td>15472.0</td>
<td>14301</td>
<td>14935</td>
</tr>
<tr>
<td>W2BL</td>
<td>D</td>
<td>2411</td>
<td>15101.8</td>
<td>14094</td>
<td>14879</td>
</tr>
<tr>
<td>W2BS</td>
<td>L</td>
<td>1941</td>
<td>1708.0</td>
<td>1534</td>
<td>1688</td>
</tr>
<tr>
<td>W2BS</td>
<td>D</td>
<td>1583</td>
<td>1683.0</td>
<td>1492</td>
<td>1670</td>
</tr>
<tr>
<td>W2CL</td>
<td>L</td>
<td>2964</td>
<td>13724.6</td>
<td>12445</td>
<td>13271</td>
</tr>
<tr>
<td>W2CL</td>
<td>D</td>
<td>2045</td>
<td>13103.5</td>
<td>11912</td>
<td>12970</td>
</tr>
<tr>
<td>W2CS</td>
<td>L</td>
<td>1666</td>
<td>1993.0</td>
<td>1600</td>
<td>1883</td>
</tr>
<tr>
<td>W2CS</td>
<td>D</td>
<td>1334</td>
<td>1902.5</td>
<td>1484</td>
<td>1806</td>
</tr>
<tr>
<td>W3AL</td>
<td>L</td>
<td>3803</td>
<td>12166.0</td>
<td>11769</td>
<td>11958</td>
</tr>
<tr>
<td>W3AL</td>
<td>D</td>
<td>2465</td>
<td>12074.5</td>
<td>11696</td>
<td>12011</td>
</tr>
<tr>
<td>W3AS</td>
<td>L</td>
<td>2030</td>
<td>1083.0</td>
<td>966</td>
<td>1037</td>
</tr>
<tr>
<td>W3AS</td>
<td>D</td>
<td>1714</td>
<td>1056.0</td>
<td>943</td>
<td>1039</td>
</tr>
<tr>
<td>W3BL</td>
<td>L</td>
<td>3623</td>
<td>14487.3</td>
<td>13257</td>
<td>13916</td>
</tr>
<tr>
<td>W3BL</td>
<td>D</td>
<td>2338</td>
<td>14194.0</td>
<td>13093</td>
<td>13800</td>
</tr>
<tr>
<td>W3BS</td>
<td>L</td>
<td>1914</td>
<td>1414.0</td>
<td>1320</td>
<td>1399</td>
</tr>
<tr>
<td>W3BS</td>
<td>D</td>
<td>1596</td>
<td>1389.0</td>
<td>1275</td>
<td>1379</td>
</tr>
<tr>
<td>W3CL</td>
<td>L</td>
<td>3016</td>
<td>12683.6</td>
<td>11352</td>
<td>12201</td>
</tr>
<tr>
<td>W3CL</td>
<td>D</td>
<td>1975</td>
<td>12093.3</td>
<td>10833</td>
<td>11929</td>
</tr>
<tr>
<td>W3CS</td>
<td>L</td>
<td>1644</td>
<td>1675.0</td>
<td>1351</td>
<td>1580</td>
</tr>
<tr>
<td>W3CS</td>
<td>D</td>
<td>1353</td>
<td>1593.0</td>
<td>1207</td>
<td>1523</td>
</tr>
</tbody>
</table>
6.4. Computational results

Table 6.4: Results of the KFRP with tabu search and parallel descent for problem instances of manufacturer M1. The meaning of the columns is the same as in Table 6.3.

<table>
<thead>
<tr>
<th>set</th>
<th>N</th>
<th>UB_{LP}</th>
<th>init</th>
<th>Tabu search</th>
<th>Parallel descent</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>max</td>
<td>avg</td>
</tr>
<tr>
<td>M1AL</td>
<td>L</td>
<td>642</td>
<td>1001.0</td>
<td>978</td>
<td>999</td>
</tr>
<tr>
<td></td>
<td>D</td>
<td>399</td>
<td>983.5</td>
<td>946</td>
<td>983*</td>
</tr>
<tr>
<td>M1AS</td>
<td>L</td>
<td>388</td>
<td>160.0</td>
<td>147</td>
<td>160*</td>
</tr>
<tr>
<td></td>
<td>D</td>
<td>282</td>
<td>155.0</td>
<td>145</td>
<td>155*</td>
</tr>
<tr>
<td>M1BL</td>
<td>L</td>
<td>620</td>
<td>1444.5</td>
<td>1397</td>
<td>1441</td>
</tr>
<tr>
<td></td>
<td>D</td>
<td>385</td>
<td>1409.0</td>
<td>1352</td>
<td>1409*</td>
</tr>
<tr>
<td>M1BS</td>
<td>L</td>
<td>374</td>
<td>245.3</td>
<td>203</td>
<td>243</td>
</tr>
<tr>
<td></td>
<td>D</td>
<td>270</td>
<td>234.8</td>
<td>203</td>
<td>234*</td>
</tr>
<tr>
<td>M1CL</td>
<td>L</td>
<td>557</td>
<td>1737.4</td>
<td>1639</td>
<td>1735</td>
</tr>
<tr>
<td></td>
<td>D</td>
<td>355</td>
<td>1676.8</td>
<td>1626</td>
<td>1675</td>
</tr>
<tr>
<td>M1CS</td>
<td>L</td>
<td>340</td>
<td>323.2</td>
<td>274</td>
<td>319</td>
</tr>
<tr>
<td></td>
<td>D</td>
<td>245</td>
<td>309.5</td>
<td>242</td>
<td>309*</td>
</tr>
</tbody>
</table>

Table 6.5: Results of the KFRP with tabu search and parallel descent for problem instances of manufacturer M2. The meaning of the columns is the same as in Table 6.3.

<table>
<thead>
<tr>
<th>set</th>
<th>N</th>
<th>UB_{LP}</th>
<th>init</th>
<th>Tabu search</th>
<th>Parallel descent</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>max</td>
<td>avg</td>
</tr>
<tr>
<td>M2AL</td>
<td>L</td>
<td>1101</td>
<td>8502.0</td>
<td>8090</td>
<td>8497</td>
</tr>
<tr>
<td></td>
<td>D</td>
<td>646</td>
<td>8414.0</td>
<td>7977</td>
<td>8412</td>
</tr>
<tr>
<td>M2AS</td>
<td>L</td>
<td>985</td>
<td>1883.0</td>
<td>1554</td>
<td>1876</td>
</tr>
<tr>
<td></td>
<td>D</td>
<td>759</td>
<td>1873.0</td>
<td>1586</td>
<td>1870</td>
</tr>
<tr>
<td>M2BL</td>
<td>L</td>
<td>1078</td>
<td>10458.0</td>
<td>8548</td>
<td>10363</td>
</tr>
<tr>
<td></td>
<td>D</td>
<td>628</td>
<td>10379.5</td>
<td>8393</td>
<td>10343</td>
</tr>
<tr>
<td>M2BS</td>
<td>L</td>
<td>980</td>
<td>2320.0</td>
<td>1763</td>
<td>2297</td>
</tr>
<tr>
<td></td>
<td>D</td>
<td>754</td>
<td>2303.0</td>
<td>1739</td>
<td>2290</td>
</tr>
<tr>
<td>M2CL</td>
<td>L</td>
<td>1014</td>
<td>10524.0</td>
<td>7989</td>
<td>10370</td>
</tr>
<tr>
<td></td>
<td>D</td>
<td>582</td>
<td>10269.0</td>
<td>7823</td>
<td>10237</td>
</tr>
<tr>
<td>M2CS</td>
<td>L</td>
<td>973</td>
<td>2600.0</td>
<td>1914</td>
<td>2599</td>
</tr>
<tr>
<td></td>
<td>D</td>
<td>738</td>
<td>2584.0</td>
<td>1874</td>
<td>2578</td>
</tr>
</tbody>
</table>

1000 seconds on a 100 MHz Pentium processor running under MS-DOS. This running time corresponds to 600 seconds on a 200 MHz PentiumPro processor running under MS-DOS.
Table 6.6: Results of the KFRP with tabu search and parallel descent for problem instances of manufacturer M3. The meaning of the columns is the same as in Table 6.3.

<table>
<thead>
<tr>
<th></th>
<th>set</th>
<th>N</th>
<th>UB_{LP}</th>
<th>init</th>
<th>Tabu search</th>
<th>Parallel descent</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>max</td>
<td>avg</td>
</tr>
<tr>
<td>M3AL</td>
<td>L</td>
<td>1650</td>
<td>33174.0</td>
<td>32365</td>
<td><strong>33124</strong></td>
<td>33096</td>
</tr>
<tr>
<td>M3AL</td>
<td>D</td>
<td>1317</td>
<td>33163.0</td>
<td>32365</td>
<td><strong>33128</strong></td>
<td>33098</td>
</tr>
<tr>
<td>M3AS</td>
<td>L</td>
<td>1268</td>
<td>3009.0</td>
<td>2867</td>
<td><strong>3007</strong></td>
<td>3001</td>
</tr>
<tr>
<td>M3AS</td>
<td>D</td>
<td>1158</td>
<td>3005.0</td>
<td>2867</td>
<td><strong>2999</strong></td>
<td>2996</td>
</tr>
<tr>
<td>M3BL</td>
<td>L</td>
<td>1620</td>
<td>31586.0</td>
<td>29329</td>
<td><strong>31471</strong></td>
<td>31445</td>
</tr>
<tr>
<td>M3BL</td>
<td>D</td>
<td>1296</td>
<td>31544.0</td>
<td>29192</td>
<td><strong>31494</strong></td>
<td>31414</td>
</tr>
<tr>
<td>M3BS</td>
<td>L</td>
<td>1237</td>
<td>3063.0</td>
<td>2684</td>
<td><strong>3037</strong></td>
<td>3031</td>
</tr>
<tr>
<td>M3BS</td>
<td>D</td>
<td>1121</td>
<td>3051.5</td>
<td>2688</td>
<td><strong>3031</strong></td>
<td>3026</td>
</tr>
<tr>
<td>M3CL</td>
<td>L</td>
<td>1544</td>
<td>20603.5</td>
<td>17128</td>
<td><strong>20474</strong></td>
<td>20451</td>
</tr>
<tr>
<td>M3CL</td>
<td>D</td>
<td>1237</td>
<td>20296.0</td>
<td>17013</td>
<td><strong>20240</strong></td>
<td>20205</td>
</tr>
<tr>
<td>M3CS</td>
<td>L</td>
<td>1137</td>
<td>2533.5</td>
<td>2033</td>
<td><strong>2490</strong></td>
<td>2474</td>
</tr>
<tr>
<td>M3CS</td>
<td>D</td>
<td>1030</td>
<td>2472.5</td>
<td>1957</td>
<td><strong>2460</strong></td>
<td>2450</td>
</tr>
</tbody>
</table>

**Discussion.** Because the distribution centers already operate with forward pick storage, the savings in the cost in the Tables 6.3 - 6.6 present no direct indication of the potential savings in handling time for the current distribution centers. Larger forward pick storage result of course in larger savings, however the return per unit of storage capacity is decreasing.

Tabu search finds for the majority of the instances the best or optimal value for the KFRP. The number of slot sizes $N$ determines if the problem instance is difficult to solve.

For each planning period of the wholesaler, tabu search could on average improve the initial solution by 9%, according to Table 6.3. However, the maximum cost remained more than 2% from the LP-bound. According to Table 6.4, the smaller problem instances of manufacturer M1 result in initial solutions that are optimal or close to the LP-bound. These solutions cannot be improved anymore with local search. Note that the problem instances are relatively small. It can be seen that the problem instances with divisible slot sizes result in all the cases, except one, in an optimal solution. From the problem instances derived with the datasets of the manufacturers M2 and M3, the initial solutions derived with the GREL algorithm can be improved with as much as 46%, according to Tables 6.5 - 6.6. The corresponding solutions found with tabu search are always within 1% of the LP-bound UB_{LP}. 
6.4. Computational results

We observe large differences when the results obtained from the dataset of the wholesaler are compared with the results obtained from the datasets of the manufacturers M1, M2 and M3. We were not able to come up with a plausible explanation for the differences in results based on the differences between the problem instances of the wholesaler and the manufacturers. From the obtained results we may conclude that the type of set of slot sizes and the type of layout have no significant effect on the performance of the solution strategy.

The effect of the type of layout on the savings may be attributed completely to the difference in cycle times between both layouts. The set of linear slot sizes seems to yield higher savings than the set with divisible slot sizes. This may perhaps be attributed to the larger number of possible slot sizes for each article and hence better opportunities for fine tuning the utilization of the forward pick storage. The application of the stock location assignment problem, which will be discussed in the next chapter, may prove whether the selected articles for the forward pick storage can actually be assigned to locations in the forward pick storage.

The model for the FRP described by Van den Berg [1996] introduces a distinction between advance replenishments to fill the forward pick storage and concurrent replenishments to refill forward pick slots during the collection of customer orders. This distinction is based on the assumption that advance replenishments can be performed during idle time. The warehouses in our case examples used the idle time for handling putaway requests and therefore cheap advance replenishments were not available. Van den Berg [1996] assumes that we always have to pick from the forward pick storage when the article has been assigned there. If we allow unit loads to be retrieved directly from the reserve storage, then we observe less concurrent replenishments and higher savings than with the assumed restriction.

Besides the proposed solution strategy for the FRP based on the knapsack problem, there exist much simpler decision rules to allocate storage space to the forward pick storage. A few of these decision rules that are frequently used in practice are mentioned by Van den Berg [1996]. The equal space procedure (ESP) allocates an equal amount of storage space to each article in the forward pick storage, which usually equals a unit load. The equal time procedure (ETP) allocates the expected demand for each article for a certain period of time to the forward pick storage. From our experiences with distribution centers for perishables we have to
conclude that the assortments are relatively large and that there is seldom ample forward pick storage capacity available for both decision rules. Forward pick storage is usually created at floor level but floor space in an insulated and air-conditioned building is more expensive than floor space in a distribution center for non-perishables.
Stock Location Assignment

In this chapter, we discuss models and algorithms for the stock location assignment problem (SLAP). After discussing the developed solution strategies, we dwell on the difference between a handling and a space oriented strategy. We present computational studies of randomly generated problem instances to illustrate the differences between both strategies with respect to the SLAP.

7.1 Solution strategies for the SLAP

All versions of the SLAP introduced in Section 3.1.7 can be modelled as a generalized assignment problem (GAP). Ross & Soland [1975] describe the formulation of the GAP as a mathematical programming model and the solution of problem instances of modest sizes by applying a branch and bound algorithm. Fischer, Jaikumar & Van Wassenhove [1986] proved that the GAP is NP-hard. Fischer, Jaikumar & Van Wassenhove [1986], Shmoys & Tardos [1993], and Cattrysse, Salomon & Van Wassenhove [1994] report among others computational studies with approximation algorithms for relatively large problem instances of the GAP. Cattrysse & Van Wassenhove [1992] published a survey of models and algorithms for the GAP. In general, the studies of the GAP reported in the literature con-
centrate on a large number of articles compared to the number of storage locations. Solutions of real world problem instances of the SLAP assign only a small number of articles to each location. Moreover, the locations have the same storage capacity due to the type of storage racks normally used in distribution centers for perishables. In this thesis we looked for models and algorithms that exploit these characteristics of the SLAP.

Recall that we can solve the SLAP separately for the forward pick storage and the reserve storage. We use in this chapter the SLAP for the forward pick storage as an example for both warehousing systems. We made the following assumptions about the problem instances of the SLAP for the forward pick storage.

- The storage capacity \( V(l) \) of a location \( l \in \mathcal{L} \) is always equal to the maximum unit load volume \( \omega_u \in \mathbb{N}^+ \).
- The allocation of storage space \( v_t(a, t) \in \mathbb{N}^+ \) to an article \( a \in \mathcal{A} \) in a planning period \( t \in T \) is less or equal to the maximum unit load volume \( \omega_u \in \mathbb{N}^+ \).
- The number of available locations \( |\mathcal{L}| \) is equal to the number of articles \( |\mathcal{A}| \).

The two last assumptions ensure feasible solutions for the SLAP since we can assign an article to at most one location in each planning period and an article always fits in a location with the assumptions above.

We start now with the investigation of the handling oriented strategy of the SLAP.

### 7.1.1 Handling oriented strategy

For a single planning period, the solution strategies for the SLAP/H are well known and extensively studied. In this case solutions of the SLAP/H can be found by applying the Cube-per-Order Index or COI rule, introduced by Heskett [1963]. Articles with the highest order frequency per unit of storage capacity are located close to the i/o point in this strategy to minimize the travel distances and therefore the expected handling time. Malette & Francis [1972] proved that ignoring the storage capacity constraints for each individual location in this procedure results in an optimal assignment of the articles to the locations. We expected that a construction algorithm for the SLAP/H based on the COI rule results in feasible solutions which can still be improved by local search. We used such a construction algorithm to generate initial solutions for the local
search.

In the computational studies of the SLAP\|H, we replaced the expected handling time with the travel time of the one way trip from the i/o point to the location of the slot. We omitted the return trip and other factors such as stop time, grab time and time for loading and unloading. These times could be ignored since they do not influence the assignment of a location to an article. In our simplified handling model the number of visits to a forward pick location of an article $a \in A$ in a planning period $t \in T$ is represented by the number of handling requests for article units $\eta(a, t)$ instead of the order frequency for article units $\gamma(a, t)$. We need a definition of the travel distance to a location from the i/o point since this variable determines the difference in handling times between the slot plans.

**Definition 7.1** (Travel distance location). The travel distance $d_l(l)$ to a location $l \in L$ gives the distance along the pick route from the i/o point to the location $l$.

The expected handling time $\Theta(a, t, l)$ associated with the assignment of an article $a \in A$ in planning period $t \in T$ to location $l \in L$ can be calculated with

$$\Theta(a, t, l) = \eta(a, t) \cdot \frac{d_l(l)}{M_f}, \quad (7.1)$$

where $M_f$ gives the walking speed of the forward picker.

**Initial solutions and lower bounds.** We were not able to find suitable construction algorithms for the multi-period SLAP with fixed assignments in literature. Because of this, we present a greedy construction algorithm that is based on the COI rule. With fixed assignments throughout the year, we have to sort the articles with respect to the sum of the number of handling requests per unit of storage space over the planning periods. The storage space allocated to an article is equal to the allocated slot size, which we determined with the KFRP in Chapter 6. The modified greedy algorithm is called the Multi-period Cube-per-Order Index (MULTICOI) algorithm.
Multi-period Cube-per-Order Index algorithm (MULTICOI)

1. Let \( A := |A| \) and \( L := |L| \). Sort the articles \( a_1, \ldots, a_A \), such that
\[
\sum_{t \in \mathcal{T}} \eta_a(a_{\phi(1)}, t)/v_t(a_{\phi(1)}, t) \geq \cdots \geq \sum_{t \in \mathcal{T}} \eta_a(a_{\phi(A)}, t)/v_t(a_{\phi(A)}, t),
\]
with \( \phi : \mathbb{N} \rightarrow \mathbb{N} \). Sort the locations \( l_1, \ldots, l_L \), such that
\[
d_i(l_{\varphi(1)}) \leq \cdots \leq d_i(l_{\varphi(L)}),
\]
with \( \varphi : \mathbb{N} \rightarrow \mathbb{N} \). Let for all \( l \in L \) and \( t \in \mathcal{T} \) \( \tilde{V}(l, t) := \omega_{\mu} \). Start with the article with the largest number of handling requests per unit of storage capacity or smallest COI by setting \( i := 1 \).

2. Start with the first location by letting \( j := 1 \). While \( j < L \) and not for all \( t \in \mathcal{T} \) \( \tilde{V}(l_{\varphi(j)}, t) - v_t(a_{\varphi(i)}, t) \geq 0 \) do \( j := j + 1 \).

3. Assign the article to the location: \( \hat{x}_t(a_{\varphi(i)}) := l_{\varphi(j)} \) and for all \( t \in \mathcal{T} \)
\[
\tilde{V}(l_{\varphi(j)}, t) := \tilde{V}(l_{\varphi(j)}, t) - v_t(a_{\varphi(i)}, t).
\]
4. \( i := i + 1 \). If \( i \leq A \), then go to 2. Otherwise stop.

The sorting of the articles in Step 1 of MULTICOI, determines the sequence in which we try to fit the slot sizes of the articles in the locations. The locations are sorted on the distance to the i/o point. An article fits in a location \( l \) when there is enough remaining storage capacity \( \tilde{V}(l, t) \) for each planning period \( t \). The assignment is finished when we reach the end of the article list.

The following theorem gives the time complexity of MULTICOI.

**Theorem 7.1.** MULTICOI is able to find a solution for the SLAP[H in \( O(T \cdot A \cdot L + A \cdot \log A + L \cdot \log L) \) time.

**Proof.** Step 1 requires \( O(A \cdot T) \) time for calculating the sums of \( \eta_a(a, t)/v_t(a, t) \). \( O(A \cdot \log A) \) time is required for sorting the sums, \( O(L \cdot \log L) \) time is required for sorting the locations and \( O(L \cdot T) \) time is required for initializing \( \tilde{V}(l, t) \). Finding a suitable location takes at most \( O(L \cdot T) \) time. Assigning the article in Step 3 can be performed in \( O(T) \) time. Steps 2 and 3 are repeated \( A \) times. Hence, the time complexity of the MULTICOI algorithm is \( O(T \cdot A \cdot L + A \cdot \log A + L \cdot \log L) \). \( \square \)
7.1. Solution strategies

The COI rule yields optimal solutions in the case of relaxation of the constraints on capacity for the individual bins and when we also allow variable assignments throughout the year. The lower bound on handling $LB_h$ is equal to the sum of the results of the COI rule for the separate planning periods.

$$LB_h = \sum_{t \in T} \sum_{a \in A} \Theta(a, t, x_f(a, t)).$$  \hfill (7.2)

We used $LB_h$ to check whether the found solution for a given problem instance is close to an optimal solution since the lower bound is always less or equal than the optimal solution.

**Neighborhood function.** As with the KFRP, we cannot guarantee feasible solutions for the SLAP|H when we assign each article randomly to a location. The procedure in MULTICOI guarantees feasible solutions for the SLAP|H. Therefore, we developed the same type of hybrid neighborhood function as described in Section 6.3.2. The neighborhood function uses a modification of MULTICOI and the original moves in the neighborhood functions of the local search techniques tabu search and parallel descent, described in Section 4.2. Instead of directly assigning articles to the locations, we assign a priority value to each article. These priority values are applied for the sorting of the articles in Step 1 of MULTICOI instead of the expected number of handling requests per unit of storage space. The moves in the neighborhood function of the local search techniques change or swap priorities. Repeated application of the moves from tabu search or parallel descent makes it possible to visit all feasible solutions in the search space. Hence, in an indirect way, the assignment of the articles to the locations is carried out by using a construction algorithm.

7.1.2 Space oriented strategy

When we focus only on storage capacity as in the SLAP|U, the problem changes into a multi-period variant of the bin packing problem. Each bin corresponds to a location with a fixed storage capacity. For each planning period, an assignment of articles to locations or bins may not exceed the storage capacity of the individual bins. In this way, the cost function of the SLAP|U is equal to the maximum number of bins needed over all the planning periods. The classic bin packing problem assumes that all bins have equal storage capacity. With different storage capacities of the
locations, we can still use the construction algorithms for bin packing, but the performance guarantees derived for these approximation algorithms no longer hold.

For the bin packing problem, the First Fit Decreasing or FFD algorithm described by Garey & Johnson [1979] is one of the most effective and efficient polynomial algorithms known. The FFD algorithm first sorts the articles on the slot size in a non-increasing order. The articles are assigned in this sequence to the locations, in the same way as MULTICOI. For bin packing problems with divisible slot sizes and a single planning period, FFD finds the optimal solution. We investigated this special case further in our research, since this property of the slot sizes frequently occurs in practice using storage racks. The sorting of the articles with respect to the slot sizes in the FFD algorithm can be omitted when the storage capacity of the bin $B$ fits in the sequence of divisible slot sizes in such a way that $s_{\text{max}}B$ holds. According to Coffman, Garey & Johnson [1987], the First Fit algorithm also finds in this case an optimal solution for problem instances with a single planning period.

Fixed assignments all the year round require more storage capacity because of a lower flexibility in storage and retrieval, and therefore a lower utilization of the storage capacity. We expect the same situation in solutions of the SLAP[H. The version of the SLAP[U with fixed assignments throughout the year for a single warehousing system resembles the vector packing problem (VPP) described by Garey, Graham, Johnson & Yao [1976]. We define the VPP in the following way.

**Definition 7.2** (Vector packing). An instance of vector packing is given by a four-tuple $I = (U, d, s, B)$, where $U$ is a finite set of slots with a $d$-dimensional vector of slot sizes $s(u) = (s_1(u), s_2(u), \ldots, s_d(u))$ with $s_i(u) \in \mathbb{N}^+$ for each $u \in U$ and $B = (B_1, B_2, \ldots, B_d)$ is a $d$-dimensional vector of positive integer bin capacities. The problem is to find a minimum number of bins, given that the contents of any bin, i.e., slots, must have vector sum of the slot sizes less than or equal to its capacity. □

Note that $s : U \to \mathbb{N}_0^d$. Coffman, Garey & Johnson [1987] proved that for the general case of the vector packing problem without the divisibility constraint, the asymptotic worst-case ratio of the First Fit algorithm is $|T| + 0.7$ and the asymptotic worst-case ratio of the First Fit Decreasing algorithm is $|T| + 1/3$. For all dimensions $|T|$ of the vector packing problem with the divisibility constraint, the asymptotic worst-case ratio of the First
7.1. Solution strategies

Fit and the First Fit Decreasing algorithm is equal to $|T|$. Up to now, these weak bounds found by Garey, Graham, Johnson & Yao [1976] still hold. We may conclude that divisible slot sizes do not guarantee optimal solutions for the vector packing problem. The described asymptotic worst-case ratios indicate that the performance of the adapted FFD algorithm decreases quickly with respect to the number of planning periods.

We developed a solution strategy based on local search for the VPP to investigate the effect of the storage capacity constraints in the SLAP|H.

**Initial solutions and lower bound.** With fixed assignments throughout the year, we have to sort the articles with respect to the maximum or the sum of the allocated stock levels per planning period. We have chosen for the sum of the slot sizes instead of the maximum, which alternative approach is described by Garey, Graham, Johnson & Yao [1976] for the vector packing problem. The reason for this is that we expected that the application of this sum results in less tie breaks in the sorted list of articles.

The following construction algorithm for the VPP is called Multi-period First Fit Decreasing (MULTIFFD), and differs only in Step 1 from the MULTICOI algorithm. The MULTIFFD algorithm sorts the articles on size instead of the value of the COI. With equal numbers of handling requests for all articles the ranking is the reverse of that from the ranking in the COI algorithm with all other steps kept the same, as is shown in the following algorithm description.

---

**Multi-period First Fit Decreasing algorithm (MULTIFFD)**

1. Let $A := |A|$ and $L := |L|$. Sort the articles $a_1, \ldots, a_A$, such that

$$
\sum_{t \in T} v_t(a_{\phi(1)}, t) \geq \cdots \geq \sum_{t \in T} v_t(a_{\phi(A)}, t),
$$

with $\phi : \mathbb{N} \to \mathbb{N}$. Sort the locations $l_1, \ldots, l_L$, such that

$$
d_1(l_{\phi(1)}) \leq \cdots \leq d_1(l_{\phi(L)}),
$$

with $\varphi : \mathbb{N} \to \mathbb{N}$. Let for all $l \in \mathcal{L}$ and $t \in \mathcal{T}$ $\tilde{V}(l, t) := \omega_u$. Start with the article with the largest size by setting $i := 1$.

2. Start with the first location by letting $j := 1$. While $j < L$ and not for all $t \in \mathcal{T}$ $\tilde{V}(l_{\varphi(j)}, t) - v_t(a_{\phi(i)}, t) \geq 0$ do $j := j + 1$. 

3. Assign the article to the location: $\tilde{x}(a_{\phi(i)}) := l_{\phi(j)}$ and for all $t \in \mathcal{T}$
   \[ \tilde{V}(l_{\phi(j)}, t) := \tilde{V}(l_{\phi(j)}, t) - v_t(a_{\phi(i)}, t). \]
4. $i := i + 1$. If $i \leq A$, then go to 2. Otherwise stop.

The time complexity of the MULTIFFD is equal to the MULTICOI algorithm.

**Theorem 7.2.** MULTIFFD does not guarantee optimal solutions for the VPP with divisible slot sizes.

*Proof by contradiction* Consider the problem instance of the VPP with dimension $d$ equal to 2 and a bin capacity $B$ equal to 4 with the following set of slots $\mathcal{U} = \{u_1, \ldots, u_6\}$ with divisible slot sizes: $s(u) = \{< 1, 1 >, < 1, 1 >, < 2, 2 >, < 2, 1 >, < 1, 2 >\}$. Sorting this set on the sum of the sizes per planning period gives the sequence $u_4 < u_5 < u_6 < u_1 < u_2 < u_3$ and results in the filling of three bins:

\[
\{
[u_4, u_5], [u_6, u_1, u_2], [u_3]\} = \\
\{[< 2, 2 >, < 2, 1 >], [< 1, 2 >, < 1, 1 >], [< 1, 1 >, < 1, 1 >] \}.
\]

The sequence $u_4 < u_1 < u_2 < u_3 < u_5 < u_6$ with the $s(u_3) \not\preceq s(u_5)$ would have resulted in only two occupied bins:

\[
\{[u_4, u_1, u_2], [u_3, u_5, u_6]\} = \\
\{[< 2, 2 >, < 1, 1 >, < 1, 1 >], [< 1, 2 >, < 2, 1 >, < 1, 1 >] \},
\]

which proves that the MULTIFFD algorithm does not guarantee optimal solutions. \[ \square \]

The same result holds for the MULTIFFD algorithm when we substitute the sum of the slot sizes with the maximum of the slot sizes, as is proposed by Garey, Graham, Johnson & Yao [1976]. Unpublished experiments that we did for our research, where we compared the proposed MULTIFFD with an MULTIFFD based on the maximum of the slot sizes, show no significant differences between both the approaches.

To check whether a solution with minimal cost is close to optimal, we used a lower bound on capacity LBc which is similar to the lower bound of the corresponding bin packing problem. With respect to this lower bound the constraints on the individual bin capacities are relaxed. Hereby we assume that all bins minus the last one are filled completely. For more
than one planning period we take the maximum required storage capacity of all planning periods.

$$\text{LB}_c = \max_{t \in T} \left\lfloor \frac{\sum_{a \in A} u_t(a, t)}{B} \right\rfloor,$$

(7.3)

where $B$ is the capacity of the bins and $\lfloor x \rfloor$ gives the first integer that is greater or equal than $x$. In the case of bins with different capacities, $B$ is the capacity of the bin with the smallest capacity.

**Neighborhood function.** We can apply the same type of neighborhood function for the VPP as for the SLAP|H. The neighborhood function uses a modification of MULTIFFD and the original moves in the neighborhood functions of the local search techniques tabu search and parallel descent, described in Section 4.2.

### 7.2 Computational results of the handling oriented SLAP

We compared the results of the solution strategy for the SLAP|H with the results of the solution strategy for the VPP using the same set of problem instances. We investigated whether the solutions of the two problems differ much in storage capacity utilization with respect to subsequently the number of planning periods, the number of articles and the type of sizes of the slots.

#### 7.2.1 Problem generation

We evaluated the solution strategies for SLAP|H and VPP using randomly generated problem instances. We have chosen for generated instances since these make it easier to analyse the difference between the two strategies. Moreover, the available real world instances resulting after the solution of the KFRP in Chapter 6 were too small to be of interest from a computational point of view. The datasets of the manufacturers had only a single planning period.

In the generated problem instances we estimate the walking speed of a forward picker by $M_f = 1 \text{ km/h}$. The facing of an unit load is taken equal to 1 m. Moreover, we assumed that all the locations in the forward pick storage are equal and have a facing of 1 m and a capacity of 1 maximum unit load volume or $\omega_u = 8$. The assumption of a single pick route along the locations leads to a straightforward calculation of the distances between the i/o point and the storage locations, i.e. $d_i(l) = l$. 

The number of handling requests for article units \( n_a(a, t) \) for each article \( a \in A \) in planning period \( t \in T \) was randomly selected from a discrete uniform distribution on the interval \([1, 1000]\). The slot sizes \( v_f(a, t) \in \mathbb{N}_0^+ \) for each article \( a \) in planning period \( t \) were randomly selected from two sets of slot sizes, introduced in Section 6.1. Dependent of the problem instance we used the set

\[
\mathcal{W}_{\text{lin}} = \{0, 1, 2, 3, 4, 5, 6, 7, 8\}
\]

with linear slot sizes and

\[
\mathcal{W}_{\text{div}} = \{0, 1, 2, 4, 8\}
\]

with divisible slot sizes. With a bin capacity \( B \) equal to 8, the additional constraint on the problem instance \( s_{\text{max}}|B \) also holds. We generated with the above mentioned parameters 50 problem instances, being all the combinations of two sets of slot sizes, five numbers of articles equal to \( A = 100, 250, 500, 1000 \) or \( 1500 \) and five numbers of planning periods equal to \( T = 1, 2, 3, 4 \) or 6.

### 7.2.2 Algorithmic performance

The results of the test with tabu search and parallel descent are given in the Tables 7.1 and 7.2. We used the implementations of tabu search and parallel descent described in Section 4.2. Table 7.1 represents the result with respect to problem instances with linear slot sizes and Table 7.2 represents the problem instances with divisible slot sizes. The results with the VPP are only presented as a reference for the results with the SLAP|H. For each problem instance, \( A \) denotes the number of articles and \( T \) the number of planning periods. The \( \text{LB}_h \) column gives the optimal cost of COI for the problem instance, which is the lower bound for handling in the SLAP|H. Each run of a local search technique started with an initial solution constructed with MULTIFFD, given in column ‘init’. The column ‘min’ gives the minimal cost and ‘acap’ the average used storage capacity of five runs. The last two columns are concerned with the solution for the VPP of the same instance, with \( \text{LB}_c \) the lower bound on capacity and ‘mcap’ the minimal cost for storage capacity. Handling cost are expressed in hours per day and are the averages over the planning periods. The storage capacities are given in the number of bins or locations. Each run took a computation time of 600 seconds on a 200 MHz PentiumPro processor running under MS-DOS.
Table 7.1: Results of the SLAP|H with tabu search and parallel descent for problem instances with linear slot sizes. For each problem instance, $A$ denotes the number of articles and $T$ the number of planning periods. The LB$_h$ column gives the lower bound for handling in SLAP|H and the 'init' column gives the value of the initial solution constructed with the MultiFFD algorithm. The column 'min' gives the minimal cost and 'acap' the average used storage capacity of five runs. The last two columns are concerned with the solution of the VPP for the same instance, with LB$_c$ the lower bound on capacity and 'mcap' the minimal cost for storage capacity.

<table>
<thead>
<tr>
<th>A</th>
<th>T</th>
<th>LB$_h$</th>
<th>init</th>
<th>Tabu search</th>
<th>Parallel descent</th>
<th>VPP</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>min</td>
<td>acap</td>
<td>min</td>
</tr>
<tr>
<td>100</td>
<td>1</td>
<td>63</td>
<td>69</td>
<td>65</td>
<td>55.0</td>
<td>66</td>
</tr>
<tr>
<td>100</td>
<td>2</td>
<td>67</td>
<td>102</td>
<td>88</td>
<td>54.6</td>
<td>89</td>
</tr>
<tr>
<td>100</td>
<td>3</td>
<td>67</td>
<td>120</td>
<td>102</td>
<td>59.0</td>
<td>102</td>
</tr>
<tr>
<td>100</td>
<td>4</td>
<td>67</td>
<td>146</td>
<td>123</td>
<td>68.4</td>
<td>124</td>
</tr>
<tr>
<td>100</td>
<td>6</td>
<td>67</td>
<td>177</td>
<td>152</td>
<td>79.0</td>
<td>152</td>
</tr>
<tr>
<td>250</td>
<td>1</td>
<td>468</td>
<td>496</td>
<td>482</td>
<td>135.2</td>
<td>484</td>
</tr>
<tr>
<td>250</td>
<td>2</td>
<td>448</td>
<td>664</td>
<td>588</td>
<td>139.0</td>
<td>597</td>
</tr>
<tr>
<td>250</td>
<td>3</td>
<td>449</td>
<td>792</td>
<td>683</td>
<td>150.6</td>
<td>690</td>
</tr>
<tr>
<td>250</td>
<td>4</td>
<td>429</td>
<td>906</td>
<td>752</td>
<td>168.4</td>
<td>771</td>
</tr>
<tr>
<td>250</td>
<td>6</td>
<td>456</td>
<td>1112</td>
<td>968</td>
<td>198.4</td>
<td>981</td>
</tr>
<tr>
<td>500</td>
<td>1</td>
<td>2020</td>
<td>2137</td>
<td>2078</td>
<td>281.0</td>
<td>2092</td>
</tr>
<tr>
<td>500</td>
<td>2</td>
<td>1762</td>
<td>2632</td>
<td>2353</td>
<td>292.0</td>
<td>2446</td>
</tr>
<tr>
<td>500</td>
<td>3</td>
<td>1812</td>
<td>3211</td>
<td>2817</td>
<td>309.8</td>
<td>2920</td>
</tr>
<tr>
<td>500</td>
<td>4</td>
<td>1765</td>
<td>3523</td>
<td>3064</td>
<td>333.0</td>
<td>3228</td>
</tr>
<tr>
<td>500</td>
<td>6</td>
<td>1765</td>
<td>4249</td>
<td>3703</td>
<td>394.0</td>
<td>3920</td>
</tr>
<tr>
<td>1000</td>
<td>1</td>
<td>7909</td>
<td>8334</td>
<td>8170</td>
<td>569.0</td>
<td>8228</td>
</tr>
<tr>
<td>1000</td>
<td>2</td>
<td>7243</td>
<td>10489</td>
<td>9810</td>
<td>588.0</td>
<td>10145</td>
</tr>
<tr>
<td>1000</td>
<td>3</td>
<td>7137</td>
<td>12398</td>
<td>11284</td>
<td>633.0</td>
<td>11898</td>
</tr>
<tr>
<td>1000</td>
<td>4</td>
<td>7018</td>
<td>13599</td>
<td>12264</td>
<td>670.2</td>
<td>13093</td>
</tr>
<tr>
<td>1000</td>
<td>6</td>
<td>6943</td>
<td>16202</td>
<td>14622</td>
<td>763.8</td>
<td>15679</td>
</tr>
<tr>
<td>1500</td>
<td>1</td>
<td>18192</td>
<td>19038</td>
<td>18845</td>
<td>861.0</td>
<td>18939</td>
</tr>
<tr>
<td>1500</td>
<td>2</td>
<td>15852</td>
<td>23771</td>
<td>22400</td>
<td>901.4</td>
<td>23224</td>
</tr>
<tr>
<td>1500</td>
<td>3</td>
<td>15384</td>
<td>26885</td>
<td>25089</td>
<td>935.4</td>
<td>26119</td>
</tr>
<tr>
<td>1500</td>
<td>4</td>
<td>15610</td>
<td>29389</td>
<td>27616</td>
<td>995.2</td>
<td>28879</td>
</tr>
<tr>
<td>1500</td>
<td>6</td>
<td>15594</td>
<td>35826</td>
<td>33315</td>
<td>1117.0</td>
<td>35040</td>
</tr>
</tbody>
</table>

7.2.3 Discussion

When we consider the results of the SLAP|H, it can be observed from Tables 7.1 - 7.2 that the tabu search seems to be superior compared to the parallel descent search technique. This is consistent with the results from the APPL|F and the KFRP in the previous chapters. Surprisingly,
Table 7.2: Results of the SLAP|H with tabu search and parallel descent for problem instances with divisible slot sizes. The meaning of the columns is the same as in Table 7.1.

<table>
<thead>
<tr>
<th></th>
<th>SLAP</th>
<th>H</th>
<th>VPP</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Tabu search</td>
<td>Parallel descent</td>
<td></td>
</tr>
<tr>
<td></td>
<td>LBₜₜ</td>
<td>init</td>
<td>min</td>
</tr>
<tr>
<td>100</td>
<td>1</td>
<td>36</td>
<td>38</td>
</tr>
<tr>
<td>100</td>
<td>2</td>
<td>30</td>
<td>52</td>
</tr>
<tr>
<td>100</td>
<td>3</td>
<td>30</td>
<td>65</td>
</tr>
<tr>
<td>100</td>
<td>4</td>
<td>32</td>
<td>82</td>
</tr>
<tr>
<td>100</td>
<td>6</td>
<td>33</td>
<td>104</td>
</tr>
<tr>
<td>250</td>
<td>1</td>
<td>261</td>
<td>266</td>
</tr>
<tr>
<td>250</td>
<td>2</td>
<td>209</td>
<td>352</td>
</tr>
<tr>
<td>250</td>
<td>3</td>
<td>203</td>
<td>422</td>
</tr>
<tr>
<td>250</td>
<td>4</td>
<td>199</td>
<td>459</td>
</tr>
<tr>
<td>250</td>
<td>6</td>
<td>200</td>
<td>596</td>
</tr>
<tr>
<td>500</td>
<td>1</td>
<td>1104</td>
<td>1113</td>
</tr>
<tr>
<td>500</td>
<td>2</td>
<td>805</td>
<td>1283</td>
</tr>
<tr>
<td>500</td>
<td>3</td>
<td>823</td>
<td>1555</td>
</tr>
<tr>
<td>500</td>
<td>4</td>
<td>821</td>
<td>1849</td>
</tr>
<tr>
<td>500</td>
<td>6</td>
<td>816</td>
<td>2263</td>
</tr>
<tr>
<td>1000</td>
<td>1</td>
<td>4487</td>
<td>4504</td>
</tr>
<tr>
<td>1000</td>
<td>2</td>
<td>3333</td>
<td>5375</td>
</tr>
<tr>
<td>1000</td>
<td>3</td>
<td>3152</td>
<td>6128</td>
</tr>
<tr>
<td>1000</td>
<td>4</td>
<td>3139</td>
<td>7072</td>
</tr>
<tr>
<td>1000</td>
<td>6</td>
<td>3136</td>
<td>8369</td>
</tr>
<tr>
<td>1500</td>
<td>1</td>
<td>10569</td>
<td>10594</td>
</tr>
<tr>
<td>1500</td>
<td>2</td>
<td>7525</td>
<td>12343</td>
</tr>
<tr>
<td>1500</td>
<td>3</td>
<td>7112</td>
<td>13989</td>
</tr>
<tr>
<td>1500</td>
<td>4</td>
<td>7029</td>
<td>15989</td>
</tr>
<tr>
<td>1500</td>
<td>6</td>
<td>7217</td>
<td>19662</td>
</tr>
</tbody>
</table>

The problem size as expressed in the number of articles \( A \) has hardly effect on the performance of the construction and local search algorithms. It seems that problem sizes up to 1500 articles are still relatively easy from a computational point of view.

On average, the initial solution found by using the MULTICOI algorithm can be improved by 10%. Note that the solutions found with the MULTICOI algorithm for the problem instances with a single planning period and divisible slot sizes were already optimal. Overviewing all the problem instances, we observe that the ratio of the minimal cost and the lower bound on handling \( \text{LB}_h \) increases linearly with respect to the number of planning periods. In this performance ratio we substituted the
unknown optimal cost with the known lower bound. We also found a linear relation between the number of planning periods and the ratio of the minimal capacity and the lower bound on capacity $LB_c$ derived with VPP, which is plotted in Figure 7.1. According to this figure, the performance ratio stays for all instances well below the asymptotic worst-case ratio. The problem instances with linear slot sizes seem to result in better solutions than the problem instances with divisible slot sizes when compared to the lower bound $LB_c$.

If we compare the capacity results of the SLAP|H with the corresponding VPP results, it can be observed that the used capacity differs 10% or less in the case of linear slot sizes and around 2% in the case of divisible slot sizes. Changing the original sequence of the articles, which was based on handling only, seems to decrease the expected handling time and the required capacity at the same time. We conclude from these results that we may apply the SLAP|H for the space oriented strategy too.
As we mentioned in Section 7.1.2, the First Fit algorithm yields optimal solutions for problem instances with a single planning period and divisible slot sizes, when the bin capacity is divisible by the maximum slot size, i.e. $s_{\text{max}} | B$. In these cases we may construct optimal solutions starting from article lists sorted on properties such as stackability and family grouping. However, further research has to be performed in order to investigate in how far the First Fit algorithm preserves these initial sorting.

The proposed construction algorithms for the SLAP do not check on the total available storage capacity. These construction algorithms for the standard case of the SLAP optimize the handling whereby it is tried to utilize the capacity as good as possible. On the other hand, the algorithms for the SLAP|U are completely focussed on capacity utilization. In the case that the construction algorithms or the subsequently following local search are not able to find a solution that uses less storage capacity, we have to adapt the slot plan. This can be derived by reallocating storage capacity between the forward pick storage and the reserve storage, by resizing the warehouse or by lowering the required minimum storage space for the articles. With all three possibilities we have to find a new solution for the SLAP based on a different problem instance. We may apply the models for the FRP to find a new allocation of storage space for the articles.

At the decision level ‘operations management’, we do not know yet if the derived slot plans have added value for the day-to-day operations. In the absence of a slot plan at the operational level, we need a storage and retrieval policy that yields both high utilization of storage capacity and low amounts of handling. The ‘random slot’ policy results according to Roll & Rosenblatt [1983] in the highest capacity utilization. Class-based storage which uses the ‘closest available slot’ policy is most widely applied since it reduces the expected handling time while maintaining a high utilization level in the warehouse. To evaluate both policies, we may apply the dynamic bin packing model which introduces arrival and departures of items in the bin packing problem. Coffman, Garey & Johnson [1987] showed that for dynamic bin packing with divisible sizes, the First Fit algorithm can be viewed as the best possible on-line algorithm. We have to verify this result with simulation experiments, particularly when we want to adapt the First Fit algorithm to a ‘closest available slot’ policy.
8

Conclusions and recommendations

We tried in this research to integrate the specific properties of perishables in the models and algorithms that deal with the slot planning problem of distribution centers and its links with the decisions concerning the layout and equipment on one hand and the decisions dealing with the operations management on the other hand. The formulation and the solution strategy of the slot planning problem focused on the seasonal production and demand of perishables, and the need for special storage conditions during the distribution process, described in the applied quality change models.

We presented a hierarchical decomposition of the slot planning problem for perishables into three subproblems: assignment of perishables to zones, forward-reserve allocation, and stock location assignment. We showed by implementation of this hierarchical planning approach in the decision support system ADEPT that the approach can be applied in practice. With computational studies we could show that we are able to solve the problem of determining the slot plan of a distribution center for perishables.

Assignment of perishables to zones. The local search techniques tabu search and parallel descent are able to find feasible and often optimal solutions for the assignment problem for perishables, despite the many constraints on storage capacity and maximum allowed relative keeping
quality loss. The number of zones that are needed in the distribution center depends mainly on the allowed level of relative keeping quality loss and the effect of temperature on the articles in the assortment. Adequate storage capacities of the different zones and the use of an effective slot plan can keep the ethylene concentration below the critical level of 1.5 ppm. For most vegetables and fruits assortments, we recommend at least three zones. A larger number of zones depends on the importance of keeping quality to the management of a distribution center for perishables.

When we can redesign the layout, we can use a simpler procedure to find slot plans with a cost lower than in the case of a fixed layout with predefined storage conditions. The exact storage capacities and storage conditions found with this procedure cannot always be implemented in practice.

A slot plan at the level of quality change groups can be used for the storage policy in a distribution center for perishables. The additional effort to make such a slot plan and integrating it into the new storage policy is relatively small compared with the benefits from reduced keeping quality loss and less use of alternative zones. The potential of the storage policy to reduce the expected keeping quality loss in practice has still to be proven, since introduction of advanced operating policies and real-time locator systems alone in the distribution center are not enough if the guidelines are not followed by the personnel.

**Forward-reserve allocation.** The proposed solution strategy for the forward-reserve allocation finds solutions close to the optimum. The heuristic for the construction of initial solutions could sometimes be improved with 46% by the local search technique tabu search. Simpler decision rules often encountered in practice cannot be used when the assortment is relatively larger and when there is not enough storage space at floor level for the forward pick storage. This situation is common for perishables due to the high cost of storage space at floor level in an insulated air-conditioned building.

**Stock location assignment.** For the case with fixed storage locations of equal storage capacity, we observed in Chapter 7 that the handling oriented approach results in solutions with a high storage capacity utilization at the same time. This effect is getting even stronger when the slot sizes and the storage rack capacity belong to a set of divisible item sizes. This observation should be an interesting guideline for the design of a distribution center, especially for seasonal products. Remark that a
lot of perishables are also seasonal.

**Wholesaler case.** The application of the decision support system prototype in the case example of the wholesaler should give the user already better insight in the current problems with respect to quality loss. It appears that the effect of high ethylene concentrations and wrong storage temperatures are being underestimated in practice. Moreover, it was observed that the solutions presented by the system did not result in a higher utilization of the storage capacity compared to the current situation. This can be attributed partly to the inaccurate historical data on the stock levels that exaggerated the utilization in the past years.

**Further research.** We hope that the models and algorithms developed for our decision support system prototype are incorporated in the near future in a warehouse management system for distribution centers for perishables.

The expert knowledge contained in the quality change models plays an important role in the acceptance of the presented solutions. The development and validation of the quality change models should therefore continue, as well as the integration of the improved models in the models and algorithms of the decision support system. The combination of perishables with non-perishables such as dry groceries in a distribution center requires only an extension of the quality change models, especially the interactions of the products by means of odors.

At the tactical level, we used static, deterministic models for a stochastic real world problem. We still have to investigate if the results of the deterministic models are meaningful for distribution centers with an increasing or a decreasing turnover. We assumed no congestion in the distribution center but this assumption becomes invalid when the required throughput increases. The same is true for changes in the assortment of the distribution center. We need to know which quality change groups in the assortment determine the number and the storage conditions in the zones.

At the operational level, the effect of an assignment plan on storage capacity utilization with dynamic storage and retrieval of seasonal products has to be investigated yet.

**Alternative applications.** The models and algorithms described in Chapters 6 and 7 can also be used in production facilities with seasonal demand. Annevelink & Broekmeulen [1993] describe the space allocation problem in pot plant nurseries which is characterized by seasonal demand
and varying production areas during the production due to the growth of the plants.

For retailers, the space allocation and article assignment are important from a marketing point of view. The opportunities for space management at the retailers are described by Van der Kind [1994]. The developed models by Lusch [1986] and Corstjens & Doyle [1981] resemble the slot planning problem on the grocery store level. A major difference is the replacement of the expected handling time in the cost function in these models with respect to the difference between price elasticity and direct product profitability (DPP) of an article.
Bibliography

BROEKMEULEN, R.A.C.M. [1996a], The Equalizer: A New Market Me-

BROEKMEULEN, R.A.C.M. [1996b], Optimisation of a distribution network for vegetables and fruits, Working paper, ATO-DLO.


CHOE, K.-I. [1990], Aisle-based order pick systems with batching, zoning and sorting, Ph.D. thesis, Georgia Institute of Technology, Atlanta, GA.


Bibliography


GLOVER, F. [1990], Tabu search: A tutorial, Interfaces 20, 74–94.

GLOVER, F., AND H.J. GREENBERG [1989], New approaches for heuristic search: A bilateral linkage with artificial intelligence, European


HACKMAN, S.T., and M.J. ROSENBLATT [1990], Allocating items to an automated storage and retrieval system, IIE Transactions 22, 7–14.


Kind, R.P. van der [1994], Opportunities for optimizations in retail, Tijdschrift voor Marketing 4, 40–47, (in Dutch).


Koster, M.B.M. de, E.S. van der Poort, and K.J. Roodebergen [1997], When to apply optimal or heuristic routing of orderpickers, Management Report 14(13), Rotterdam School of Management.


Lusch, R.F. [1986], The new algebra of high performance retail management, Retail Control 9, 15–35.


Meffert, H.F.Th. [1990], Economic developments pertinent to chilled


ROLL, Y., AND M.J. ROSENBLATT [1983], Random versus grouped storage policies and their effect on warehouse capacity, Material Flow 1, 199–205.


VAESSENS, R.J.M., E.H.L. AARTS, AND J.K. LENSTRA [1996], Job


**List of symbols**

\[ \mathcal{A} \] set of articles

\[ \mathcal{F} \] set of forward pick slots

\[ \mathcal{G} \] set of combined reserve and forward locations

\[ \mathcal{L} \] set of locations

\[ \mathcal{Q} \] set of quality change groups

\[ \mathcal{R} \] set of reserve slots

\[ \mathcal{S} \] set of all sets of slots to which an article can be assigned

\[ \mathcal{T} \] set of planning periods

\[ \mathcal{Z} \] set of zones

\[ l_s \] location of a slot

\[ q_a \] quality change group of an article

\[ z_l \] zone of a location

\[ z_s \] zone of a slot

\[ \omega_u \] maximum unit load volume

\[ \bar{p} \] average mass per unit volume

\[ \beta \] number of article units on a unit load of an article

\[ \gamma \] number of article units that are ordered of an article in a planning period

\[ \gamma_a \] number of article units that are picked as article units for an article in a planning period

\[ \gamma_u \] number of unit loads that are picked as unit loads for an article in a planning period

\[ \eta \] total number of handling requests for an article in a planning period

\[ \eta_a \] total number of handling requests that pick article units for an article in a planning period

\[ \eta_u \] total number of handling requests that pick unit loads for an article in a planning period
\(w\) minimum required total storage space for an article in a planning period

\(w_f\) minimum required forward pick storage space for an article in a planning period

\(w_q\) group storage space requirement for a quality change group in a planning period

\(\hat{w}_q\) amount of a quality change group that is stored overnight in a zone

\(v\) storage capacity of a slot

\(V_s\) storage capacity of a set of slots

\(v_f\) total forward pick storage space allocated to an article in a planning period

\(v_r\) total reserve storage space allocated to an article in a planning period

\(V\) total storage capacity of a location

\(V_f\) forward pick storage capacity of a location

\(V_z\) total storage capacity of a zone

\(\hat{V}\) total storage capacity of the warehouse

\(\hat{V}_f\) total storage capacity of the forward pick area in the warehouse

\(\hat{V}\) remaining storage capacity of a location or a zone

\(\Theta\) handling time for an article in a set of slots in a planning period

\(\Theta_{fo}\) handling time for an article with only forward pick slots in a planning period

\(\Theta_{ro}\) handling time for an article with only reserve pick slots in a planning period

\(\Theta_{rf}\) handling time for an article with both reserve and forward pick slots in a planning period

\(\Theta_{save}\) potentially saved handling time for an article in a planning period

\(\Theta_{fpk}\) total time for forward picking of an article in a set of slots in a planning period

\(\Theta_{ptw}\) total time for putaway of an article in a set of slots in a planning period

\(\Theta_{rpk}\) total time for reserve picking of an article in a set of slots in a planning period
List of symbols

$\Theta_{rpl}$ total time for replenishment of an article in a set of slots in a planning period

$\theta_{gaf}$ grab time for an article unit of a forward picker

$\theta_{gar}$ grab time for an article unit of a reserve picker

$\theta_{gur}$ place/extract time for a unit load of a reserve picker

$\theta_{fix}$ fixed cycle time

$\theta_{cfr}$ pick cycle time to a forward pick slot of a forward picker

$\theta_{cfr}$ pick cycle time to a forward pick slot of a reserve picker

$\theta_{crr}$ pick cycle time to a reserve slot of a reserve picker

$\bar{\Theta}_{rpl}$ time for a single replenishment

$D$ the length of the warehouse

$d_s$ travel distance to a slot

$d_f$ travel distance to a forward pick slot

$d_r$ travel distance to a reserve slot

$d_l$ travel distance to a location

$M_f$ travel speed for a forward picker

$M_r$ travel speed for a reserve picker

$k$ static remaining keeping quality of a quality change group

$k_e$ effect of ethylene on a quality change group in a zone for a specific ethylene concentration

$k_h$ effect of handling on a quality change group in a zone

$k_{ref}$ reference keeping quality of a quality change group in a zone

$\varepsilon_{ea}$, $\varepsilon_{eb}$ ethylene sensitivity of a quality change group

$\varepsilon_{ep}$ ethylene production of a quality change group

$\varepsilon_h$ maximum handling sensitivity of a quality change group

$\varepsilon_{pa}$, $\varepsilon_{pb}$ the first quality change process of a quality change group

$\varepsilon_{qa}$, $\varepsilon_{qb}$ the second quality change process of a quality change group

$\Lambda$ relative keeping quality loss of a quality change group in a zone with a specific ethylene concentration

$\Lambda_{max}$ maximum allowed relative keeping quality loss of a quality change group

$\lambda$ maximum relative keeping quality loss for all articles that stay overnight in the warehouse

$\bar{\lambda}$ average of $\lambda$ over the year

$\lambda_{max}$ maximum of $\lambda$ over the year

$\varepsilon_p$ ethylene production of a quality change group in a zone

$\varepsilon$ ethylene concentration in a zone
List of symbols

\( \tau \) \hspace{1cm} \text{temperature in a zone}

\( \tau_{\text{ref}} \) \hspace{1cm} \text{threshold temperature of a quality change group above which the first quality change process takes place}

\( \zeta \) \hspace{1cm} \text{total volume of a zone}

\( \kappa \) \hspace{1cm} \text{number of times per day a zone is completely refreshed through ventilation}

\( \sigma \) \hspace{1cm} \text{accessibility of a zone}

\( \omega_h \) \hspace{1cm} \text{weightfactor for handling}

\( \omega_u \) \hspace{1cm} \text{weightfactor for storage capacity utilization}

\( \omega_q \) \hspace{1cm} \text{weightfactor for keeping quality}

\( \pi_c \) \hspace{1cm} \text{penalty for exceeding the storage capacity}

\( \pi_q \) \hspace{1cm} \text{penalty for exceeding the maximum allowed keeping quality loss}

\( \chi \) \hspace{1cm} \text{assignment of an article in a planning period to a set of slots}

\( x_f \) \hspace{1cm} \text{assignment of an article in a planning period to a forward pick slot location}

\( x_r \) \hspace{1cm} \text{assignment of an article in a planning period to a reserve slot location}

\( \hat{x}_f \) \hspace{1cm} \text{fixed assignment of an article to a forward pick slot location throughout the year}

\( \hat{x}_r \) \hspace{1cm} \text{fixed assignment of an article to a reserve slot location throughout the year}

\( x_u \) \hspace{1cm} \text{assignment of storage space of an article in a planning period to the forward pick area}

\( y \) \hspace{1cm} \text{utilization of a location in a planning period}

\( x_z \) \hspace{1cm} \text{assignment of a quality change group in a planning period to a zone}

\( \hat{x}_z \) \hspace{1cm} \text{year round assignment of a quality change group to a zone}

\( \tau_z \) \hspace{1cm} \text{assignment of a zone to a temperature}

\( V_z' \) \hspace{1cm} \text{allocation of storage capacity to a zone}

\( \zeta_z \) \hspace{1cm} \text{allocation of total volume to a zone}

\( \text{LB}_c \) \hspace{1cm} \text{lower bound on storage capacity for SLAP\text{U}}

\( \text{LB}_h \) \hspace{1cm} \text{lower bound on handling for SLAP\text{H}}

\( \text{LB}_q \) \hspace{1cm} \text{lower bound on the keeping quality loss in APP}

\( \text{UB}_{LP} \) \hspace{1cm} \text{upper bound for KFRP}
Index

accessible, 15, 21
accumulation, 21, 38
aisle, 23
APP, 71
article, 3
    complimentary, 32
assortment, 3
commodity, 14
congestion, 27, 161
consignment, 2, 21
cross docking, 21, 40
decision support system, 6, 161
decoupling point, 18
demand dependency, 25
direct delivery, 21, 40
distribution
    center, 2
    chain, 2, 19
        integrated, 18
    function, 2
equipment
    handling, 2, 24, 38, 130
    storage, 2, 23, 128
ethylene, 15, 64
    concentration, 65
FIFO, 26, 122
FRP, 48
GAP, 145
groupage, 2, 21
handling, 2
effect on keeping quality, 15, 66
    request, 22, 23, 41
time, 38
i/o point, 23
keeping quality, 14
    initial, 15, 68
    loss, 3
    reference, 62, 63
    relative, 66
    static, 14, 62
KFRP, 52
layout, 3, 22, 129
location, 2, 53
    forward pick, 21
    reserve, 21
locator, 24
odor, 15, 64
order
    customer, 2, 21, 40
    frequency, 41
penalty method, 86
pick
    cycle, 23
    list, 24
pickers, 21
picking
    batch, 28
    forward, 21
    multi-address, 28
    of orders, 21
reserve, 21
single-address, 28, 38
planning period, 3
policy
  operating, 3
  retrieval, 2
  storage, 2, 26, 38
product, 14
putaway, 21
QAP, 83
quality change
  group, 17, 61, 89
  models, 17, 61
rack, 23, 146
replenishment, 22

shelf life, see keeping quality
shipment, 2
SLAP, 49
slot, 2
  data, 36
  forward pick, 21
  plan, 6, 36
  reserve, 21
  size, 51, 128
sorting, 29, 158
stock level, 25, 61, 91
storage
  accommodation, 2, 22
  capacity, 36
  conditions, 4, 14, 22, 62
  forward pick, 21
  policy, 122
  reserve, 21
  room, 4
  service level, 25
  space, 2, 36
  time in, 15, 20, 25, 68
supplier, 21

temperature, 15, 63
throughput, 24, 161
turnover, 161
  rate, 25
uniform article coding, 20
unit load, 21, 40
utilization
  effect on keeping quality, 15
value added logistics, 3
ventilation, 15
warehouse, 2
warehousing
  function, 2
  system, 3, 20
wholesaler, 18, 29
WMS, 7, 24
WS3P, 69
WSPP, 47
zone, 22, 60, 90
Samenvatting

In het proefschrift wordt een aanpak gepresenteerd, gebaseerd op een hiërarchische beslissingsstructuur die de efficiëntie en de effectiviteit van de primaire processen van een distributiecentrum voor bederfelijke producten verbetert. De primaire processen worden in het geval van bederfelijke producten zoals groenten en fruit bemoeilijkt door de sterke seizoensfluctuaties in productie en vraag en door de behoefte aan specifieke opslagcondities voor de verschillende producten.

De specifieke eigenschappen van de distributie van bederfelijke producten worden beschreven in Hoofdstuk 2. In het geval van agrarische producten speelt een distributiecentrum een bijzondere rol als ontkoppelpunt tussen de vaak kleinschalige, grond- en klimaatgebonden productie enerzijds en de van dag tot dag wisselende vraag van consumenten anderzijds. In tegenstelling tot niet-bederfelijke producten kan een distributiecentrum voor bederfelijke producten slechts in beperkte mate optreden als een buffer tussen vraag en aanbod door de relatief snel afnemende productkwaliteit. De verblijfstijd van de producten in het distributiecentrum is daarom relatief kort. Door het creëren van verschillende koelcellen of zones in het distributiecentrum met specifieke opslagcondities zoals temperatuur kan het verlies van houdbaarheid van een groot aantal producten tijdens de korte opslag worden gereduceerd. Door het grote aantal verschillende producttypen waarin de bederfelijke producten zijn te onderscheiden is het in de praktijk niet mogelijk om elk producttype een eigen zone toe te wijzen. Aan een zone worden daarom verschillende producttypen toegewezen en de inhoud van een zone wisselt met de seizoenen door fluctuaties in vraag en aanbod. De opslagcondities in een zone zijn een compromis tussen de gewenste opslagcondities van de verschillende producttypen. Daarnaast moet bij het toewijzen rekening worden gehouden met de interacties tussen de producten. Geuren en plantenhormonen zoals ethyleen zijn voorbeelden van stoffen die door het ene product worden geproduceerd en die bij een ander product tot verminderte houdbaarheid of productkwaliteit kunnen leiden.
De beslissingen aangaande de inrichting en besturing die het management van een distributiecentrum voor bederfelijke producten moet nemen zijn te onderscheiden in:

- Lange termijn beslissingen over het ontwerp van het distributiecentrum en over de te gebruiken hulpmiddelen zoals opslagsystemen en vorkheftrucks.

- Een locatieplan voor de middellange termijn dat voor elk product in het assortiment bepaalt waar, wanneer en hoeveel ruimte wordt gereserveerd in het distributiecentrum.

- Besturingsregels voor de korte termijn die onder andere adviseren waar een product kan worden ingeslagen en welk product kan worden uitgeslagen.

Het onderzoek richtte zich voornamelijk op het opstellen van een tactisch locatieplan, gebaseerd op historische data. In Hoofdstuk 3 wordt het probleem van het bepalen van een locatieplan opgesplitst in een aantal deelproblemen die afzonderlijk beter zijn te modelleren en aan te pakken.

We gebruiken verschillende lokale-zoekmethoden om de deelproblemen op te lossen en dus het uiteindelijke inrichtings- en locatieplan op te stellen. Deze methoden worden geïntroduceerd in Hoofdstuk 4.

In Hoofdstuk 5 wordt onder meer onderzocht hoe een locatieplan op het niveau van producttypen afhangt van het aantal zones en de opslagconditions in die zones. Het effect van een gekozen inrichting en een bijbehorend locatieplan op diverse inslagstrategiën is bestudeerd met een simulatiemodel van de primaire processen in een distributiecentrum. Een inslagstrategie die rekening houdt met het verwachte verlies van houdbaarheid blijkt bij een goede benutting van de opslagcapaciteit ook het houdbaarheidsverlies te beperken.

Het deelprobleem van de verdeling van de beschikbare opslagruimte voor een product tussen de bulk- en de grijpvoorraad is bekeken in Hoofdstuk 6. Grijpvoorraad op vloerniveau wordt gebruikt om kleine bestellingen efficiënt te verzamelen en bulkvoorraad dient om de grijpvoorraad aan te vullen en om grote eenheden te verzamelen. De ontwikkelde methode vindt bijna-optimale oplossingen die de schaarse ruimte voor grijpvoorraad zodanig benutten dat de hoeveelheid arbeid wordt geminimaliseerd.

Het deelprobleem van het bepalen van de positie van de bulk- en de grijpvoorraad van een product is onderzocht in Hoofdstuk 7. Het blijkt dat de strategie die streeft naar een minimale rijtijd voor in- en uitslag tegelij-
Samenvatting

kertijd ook de opslagcapaciteit goed benut. Indien het locatieplan vastligt voor een heel jaar, dan neemt de benutting van de opslagcapaciteit af door de slechte afstemming van het locatieplan met de wisselende voorraadhoogten.

De gepresenteerde aanpak is onder andere toegepast op een distributiecentrum voor groenten en fruit in Nederland om aan te tonen dat de ontwikkelde technieken en oplossingsstrategiën werken in de praktijk. Het geheel is geïmplementeerd in de vorm van een beslissingsondersteunend software systeem dat kan worden uitgevoerd op een standaard PC.
Curriculum Vitae

Rob Broekmeulen was born on June 6, 1961, in Beek, Limburg, the Netherlands. In Maaseik, Belgium, he took his Modern Humanities examination at the Holy Cross College in 1979. In the same year he started his study Molecular Sciences at the Agricultural University in Wageningen. He did his six months internship at the pharmaceutical division of Ciba-Geigy AG, Basel, Switzerland. In September 1986 he graduated, with Molecular Genetics and Industrial Microbiology as main subjects, and Operations Research as secondary subject.

After his graduation he worked for four years as assistant professor at the Department of Mathematics, Section Operations Research of the Agricultural University, Wageningen. Subsequently, he worked as senior researcher at the Agrotechnological Research Institute ATO-DLO till September 1997. This thesis is the result of the research that was carried out at ATO-DLO, under the supervision of prof.dr. E.H.L. Aarts and prof.dr. A.G. de Kok. He is currently working as assistant professor at the Department of International and Distribution Logistics of Eindhoven University of Technology.