Universität Bayreuth

Fakultät für Mathematik, Physik und Informatik

Lehrstuhl für Wirtschaftsmathematik
Prof. Dr. Jörg Rambau

Dissertation

zur Erlangung des Grades
„Doktor der Naturwissenschaften“
an der Universität Bayreuth

Integrated Size and Price Optimization for a fashion retailer

vorgelegt von: Dipl. Math. Miriam Kießling
Lehrstuhl für Wirtschaftsmathematik
Universität Bayreuth
95440 Bayreuth
Tel.: 0921/55-7354
miriam.kiessling@uni-bayreuth.de

vorgelegt am: 25. September 2012

Betreuer: Herr Prof. Dr. Jörg Rambau

Zusammenfassung

Die vorliegende Arbeit ist das Ergebnis einer mehrjährigen Zusammenarbeit mit einem deutschen Textildiscounter.

Das Ziel war die Entwicklung eines entscheidungsunterstützenden Systems für die Belieferung der etwa 1300 Filialen in Deutschland.

Diese weist einige Besonderheiten auf: Die Filialen werden mit vorverpackten Kombinationen von Größen eines Artikels, sogenannten Lot-Typen, beliefert. Durch die Zusammenstellung dieser Lot-Typen, die bereits in dem Niedriglohnland erfolgt, in dem die Ware produziert wird, sollen die Handgriffe im Zentrallager und damit die Kosten in Deutschland reduziert werden. Um auch den weiteren Aufwand im Zentrallager möglichst klein zu halten, werden nur maximal vier bis fünf verschiedene Lot-Typen für einen Artikel verwendet. Außerdem wird pro Artikel jede Filiale nur mit einem Lot-Typ in einer Vielfachheit beliefert.

Da es sich um Modeartikel handelt, die in der Regel nicht nachbestellt werden können, ist die Popularität des jeweiligen Produkts von vornherein unbekannt. Bedarfe können nur sehr grob, das heißt durchschnittlich und auf Warengruppenebene, geschätzt werden. Über- und Unterbelieferungen lassen sich nicht vermeiden.

Eine Einflussnahme auf den Verkaufsprozess ist durch Preisreduzierungen möglich. Um Überbelieferungen zu kompensieren, kann der Preis eines Artikels wöchentlich auf vordefinierte vom Startpreis abhängige Preisstufen herabgesetzt werden. Preisreduzierungen erfolgen für einen Artikel in allen Filialen und für alle Größen simultan.

In Rahmen der Kooperation wurden mathematische Problemformulierungen entwickelt, mit dem Zweck, Kosten für die Abweichung von Belieferung und geschätztem Bedarf zu minimieren. Der eigentliche Verkaufsprozess wurde bei der Ermittlung dieser Kosten nicht oder nur sehr grob betrachtet.

Wir beziehen nun die Möglichkeit von Preisreduzierungen bereits bei der Entscheidung über die Belieferung ein. Das Ergebnis ist das zweistufige stochastische Programm ISPO: Die sogenannte Erststufenentscheidung ist die Festlegung einer Belieferungsstrategie, die Zweitstufenentscheidung oder der Rekurs, die Entscheidung über Preisreduzierungen im Verkaufsverlauf. ISPO liefert eine ertragsmaximierende Belieferungsstrategie sowie sich darauf beziehende optimale Preisreduzierungsstrategien für betrachtete Szenarien.

ISPO ist zu komplex um es mit Standardverfahren zu lösen. Die Entwicklung von speziellen Lösern war notwendig. Zum einen präsentieren wir einen exakten Löser zum Benchmarking und zum anderen eine schnelle Heuristik für den praktischen Einsatz beim Industriepartner.

Der exakte Löser basiert auf der Idee mögliche Preisreduzierungsstrategien zu enumerieren. Damit kann ISPO auf eine frühere Problemformulierung zur Optimierung der Belieferungsstrategie, die mit Standardverfahren gelöst werden kann, zurückgeführt werden.

In der Praxis ist eine Lösung von ISPO nur durch Enumeration aller möglichen Preisreduzierungsstrategien zeitlich unmöglich.

Daher wird die Idee zu einem pro- blembezogenen Branch&Bound Verfahren erweitert. In diesem Zusammenhang ent- wickeln wir duale Schranken für allgemeine zweistufige stochastische Optimierungs- probleme, die auf der sogenannten *wait-and-see solution* aus der stochastischen Optimierung basieren. Wir zeigen, dass unsere Schranken im Allgemeinen schärfer sind.

Die Heuristik sucht ausgehend von einer zulässigen Zweitstufenentscheidung eine dazu optimale Erststufenentscheidung und alterniert dann bis zur Konvergenz zwischen zweiter und erster Stufe. Die Optimalitätslücke ist klein genug um einen praktischen Einsatz zu rechtfertigen.

In der Praxis werden die bezüglich ISPO optimalen Preisreduzierungsstrategien nicht verwendet. Stattdessen werden aktuelle Verkaufszahlen ausgenutzt. Gemäß diesen und einer angepassten Bedarfsschätzung wird wöchentlich eine neue optimale Preisreduzierungsstrategie für den verbleibenden Verkaufszeitraum ermittelt. Dafür präsentieren wir einen Algorithmus, der auf dynamischer Programmierung beruht und nicht optimale Lösungen durch sogenannte Dominanztests von vornherein auszuschliessen versucht.

ISPO, genauer gesagt unsere Heuristik, zusammen mit der wöchentlichen Aktualisierung der Preisreduzierungsstrategien bildet unser entscheidungsunterstützendes System zur integrierten Größen- und Preisoptimierung DISPO.

Wir testeten DISPO in einem fünfmonatigen Feldversuch, durchgeführt als statistisches Experiment, beim Praxispartner. Hierbei wurden Paare ähnlicher Filialen miteinander verglichen: In einer Filiale, der Testfiliale, wurde die von ISPO vorgeschlagene Belieferungsstrategie umgesetzt und wöchentlich, wie oben beschrieben, die Preisreduzierungsstrategie aktualisiert. In der anderen Filiale, der Kontrollfiliale, wurde ein früheres Modell zur Festlegung der Belieferungsstrategie eingesetzt, in dem der Verkaufsprozess nicht integriert ist. Preisreduzierungen in den Kontrollfilialen wurden vom Projektpartner angeordnet. In den Testfilialen, für die DISPO eingesetzt wurde, erzielten wir einen um mehr als $1,5$ Prozentpunkte höheren realisierten Rohertrag als in den Vergleichsfilialen.

Abstract

This thesis is the result of a collaboration with a German fashion retailer which lasted for several years. The aim was the development of a decision-support system for the supply of the about 1300 branches in Germany.

There are some specialties about the situation at our industrial partner: The branches are supplied by prepackaged size-assortments of a product which we call lot-types. With the objective to economize handling cost, these lot-types are already composed at the respective low-wage country where the article is also produced. The expense at the German central warehouse is further reduced by allowing only four or five different lot-types for the delivery of one product. Moreover, each branch is supplied by a certain quantity of a single lot-type.

For the most fashion articles replenishment is not possible. The sales success of a product is a priori unknown. Historical sales data can only be used on a higher aggregation level, e.g., the average historical demand on the commodity group level. Demand estimation is therefore very vague. Under- and oversupplies are unavoidable. Influence over the sales process is possible by marking down prices. To compensate for an oversupply of a product, weekly the price can be reduced to predefined price steps which depend on the starting price of the product. Mark-downs for an article are performed simultaneously for all branches and sizes.

Within the cooperation mathematical problem formulations with the aim to minimize measures for the deviation of supply from estimated demand had been developed. In these measures the selling process is not or only very vaguely regarded.

Now we include the possibility of marking down prices during the selling time already when deciding on the supply. The result is the two-stage stochastic program ISPO: The so-called first stage decision is the determination of a supply policy. The second stage decision, or recourse, is the decision on mark-downs during the selling time. ISPO yields an expected revenue maximizing supply strategy and corresponding optimal mark-down strategies for the considered scenarios.

ISPO it too complex to solve it via standard approaches. Customized methods had to be devised to solve ISPO. On the one side we present an exact solver for benchmarking. On the other side a fast heuristic was developed for practical use at our partner. The basic idea of our exact solver is to enumerate all possible mark-down strategies. With this it is possible to reduce ISPO to a former formulation for the optimization of supply, which can be solved via standard approaches.

In practice enumeration of all valid mark-down strategies for the purpose of solving ISPO is for reasons of time impossible. Therefore the idea is extended to a customized Branch&Bound approach. In this context we derived dual bounds for general two-stage stochastic programs which are based on the so-called wait-and-see solution from stochastic programming. We show that in general our bounds are tighter.

The heuristic, beginning with a valid second stage decision, determines an optimal

first stage decision and alternates between solving the first stage and the second stage until convergence is reached. The optimality gap is small enough to justify a practical use at the industrial partner.

In practice the by ISPO proposed mark-down strategies are not applied; instead latest sales figures are exploited. According to these and an updated demand estimation weekly a new optimal mark-down strategy for the remaining selling time of the product is determined. For this purpose we propose an algorithm which relies on dynamic programming and tries to exclude non-optimal solutions a priori by dominance checks.

ISPO, more precisely our heuristic approach, together with the weekly adaption of the mark-down strategy forms our decision support system for integrated size and price optimization DISPO.

We tested DISPO in a five-month field study, performed as a statistical experiment, at our partner where pairs of similar branches were compared. At one branch of each pair, the test branch, supply and mark-down decisions came from ISPO. With respect to latest sales figures the mark-down decisions were weekly updated via our dynamic programming approach. At the other branch, the control branch, these decisions were not integrated: Supply was determined according to a strategy resulting from a former model that disregarded the selling process and mark-downs were handled manually by our partner. For the branches at which the decisions of ISPO were implemented an average raise of 1.5 percentage points of relative revenue was observed.

Contents

List of Symbols

$\mathbb{E}$	expected value 15
S	set of sizes 7
B	set of branches 7
L	set of lot-types 7
$v^{\min}$	minimum number of items per size per lot-type 7
$v^{\max}$	maximum number of items per size per lot-type 7
$vl^{\min}$	minimum number of items per lot-type 7
$vl^{\max}$	maximum number of items per lot-type 7
M	set of multiplicities 7
$m^{\max}$	maximum multiplicity 7
κ	maximal number of supplied lot-types 7
$\underline{I}$	lower bound for overall supply 7
$\overline{I}$	upper bound for overall supply 7
$d_{b,s}$	dependent demand for Size s in Branch b 7
I	overall supply 8
$\mathrm{dist}^{\mathrm{LDP}}_{b,\ell,m}$	objective coefficients of the LDP 8
$x_{b,l,m}$	binary variable, Is Branch b supplied by Lot-type l in Multiplicity m? 8
y_l	binary variable, Lot-type l at least once delivered? 8
z_i	binary variable, at least i different lot-types delivered? 11
δ_i	opening cost for the ith used lot-type 10
pcost	pick cost 10
$d^e_{b,s}$	dependent demand for Size s in Branch b 10
ap	acquisition price 10
$\pi_{p_{\max}}$	salvage value 10
π_0	starting price 10
$\mathrm{Prob}(e)$	probability for occurrence of Scenario e 10
K	set of sales periods 15
$k_{\max}$	sellout period 15
k_{obs}	number of observation periods 15
P	set of price indices 15
π_p	price related to Price index p 15
ρ	discounting factor 15
$d^e_{k,p,b,s}$	dependent demand for Scenario e of Size s in Branch b for Price π_p in Period k 15
$u^e_{k,p}$	binary variable, price set to π_p in Period k for Scenario e? 15
$v^e_{k,b,s}$	fractional variable, stock level for Size s in Branch b at Period k for Scenario e 15

Chapter 1

Introduction

In clothing retailing mark-downs of prices and non-compliant supplies in terms of the sizes affect the revenues in a great part because of the short selling times. Supply and mark-down strategies interact substantially. In these thesis we consider these effects in an integrated mathematical model with the aim to develop a real-world revenue-maximizing decision system.

This work is motivated by more than a five years long cooperation with a German fashion retailer.

Our industrial partner supplies its about 1300 branches in Germany by prepackaged size-assortments of a product, so-called *lot-types*. A *lot-type* specifies for each size the number of items in that size in the prepackage.

With the objective to economize handling cost, these lot-types are already composed at the respective low-wage country where the related article is also produced. The expense at the German central warehouse is further reduced by supplying each branch only with a certain quantity of a single lot-type. At the maximum four or five different lot-types are used for the delivery of one product.

Thus, the supply for the particular branches and sizes can not be decided independently from each other.

Depending on the popularity of the particular article during the sales process mark-downs are performed with the aim to maximize the realized overall revenue. Prices are marked-down for a product in all branches and all sizes simultaneously.

After the regular selling time which as a rule amounts to three months, the products stay in the branches – from the clothes hanger they are stowed away on dump bins and sold for low price. According to what our industrial partner says nearly all items of a product can be sold – but a bad seller may require drastic mark-downs.

The described process is depicted in Figure 1.1. At first we take our concentration on the original process at our industrial partner. The purchasing agent decides on a quantity of the corresponding article. This quantity is based on the availability of the product and the agent's own empirical values according to how he estimates the popularity of the article. After the purchase on the amount of items per branch and size based on lot-types is decided. At a determined date the sales process for the product starts in all supplied branches simultaneously. If the product turns out to be less popular than estimated mark-downs may be disposed which are performed by the sales personnel at the branches by adapting the price tag. Moreover the new price is deposited at the system. The price steps to which the prices can be deduced are predefined and depend on the starting price of the article. The decision if and what kind of mark-down is

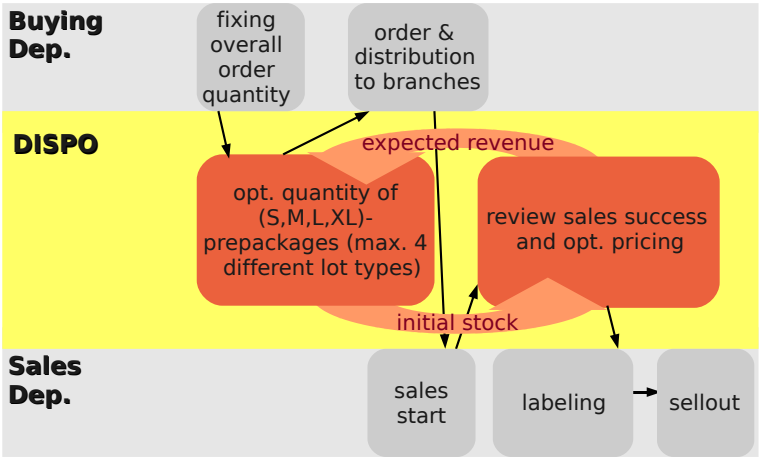

Figure 1.1: Integration of DISPO into the business process

performed is made each sales week again by the sales department on the base of latest sales figures.

Our decision support system for Integrated Size and Price Optimization DISPO will intervene at two points. When the overall supply is known DISPO will propose a supply policy based on lot-types. The specialty is that already at this point possible mark-downs are regarded. This is done by solving a stochastic program where mark-downs act as compensation for wrong supply – more exactly oversupply. Thus, with taking into account pricing already in the supply process we hope to increase the realized revenue. We call this part of DISPO also *size optimization*. The related integrated two-stage stochastic model is called ISPO.

Analogously to the original approach at our partner DISPO also intervenes during the selling time. After an observation time of two weeks, based on latest sales figures the mark-down strategy is weekly updated. We call this part of DISPO also *price optimization*.

1.1 Related Work

Linking of inventory and dynamic pricing decisions among others has been attacked in [BdB05, CSSLS04, FH99, Net06]. Common to those results is the optimal control approach via the dynamic programming approach and/or fluid approximation where discrete events resp. distributions are approximated by their expected values and such the random process is approximated by a continuous deterministic process. More recent approaches consider robustness considerations [AP06] or game theoretic aspects, like competition and equilibria [AP10]. The real-world settings of companies usually involves additional side-constraints and costs. In our case it is the restriction on the number of used lot-types and lot-type handling and opening costs that would lead to very large state spaces in dynamic programming.

Dynamic pricing is a well-studied problem in the revenue management literature (see, e.g., [BC03, GvR94, GvR97, Mon05, ZZ00] as examples). Again, complicated operational side-constraints are usually neglected in favor of a more principle study of isolated aspects. Again, some work has been done from a game theoretic point of

view, like strategic customers who are forward-looking customers which for example include possible mark-downs of prices in their buying decision (see, e.g., [YAPT09] or [MZ12]).

Classical inventory management research is less related to our topic, since there most policies deal with the optimal way to replenish stock. In our environment, no replenishment is possible.

The first steps in capturing the operations side constraints posed by the lot-based supply [GKR10] – the Lot-type Design Problem LDP – did not take pricing into account, but estimated the consequences of inventory decisions by a distance measure between supply and an estimated demand (without reference to pricing).

The LDP was extended to the Stochastic Lot-type Design Problem SLDP by regarding monetary costs and different sales scenarios [KKR11a]. The SLDP will serve as a template for our model of the size optimization stage. Since the number of possible lot-types can be very large which leads to high computation times, a Branch&Price algorithm was presented in [KKR11a].

For evaluating field studies in terms of applying different supply policies methods which are based on the comparison of sales speeds of different sizes were proposed [KKR12].

Some topics of this thesis concerning the model for ISPO, the exact and the heuristic approach to solve it and the concluding field-study at our industrial partner are treated in similar fashion in [KKR11b].

1.2 Our contribution

We present an inventory and dynamic pricing problem of a real-world fashion retailer with a set of operational side-constraints that has been unstudied so far. For this problem, we contribute a decision support system. We present the new two-stage stochastic model ISPO for optimization of supply. To solve ISPO we propose an exact Branch&Bound algorithm for benchmarking with new dual bounds based on the wait-and-see solution from stochastic programming and a fast heuristic for practical use. Moreover we devised dominance rules especially for the so-called Price Optimization Problem as it is weekly solved during the selling time. Under use of these rules we implemented a dynamic programming approach as a label setting algorithm. Moreover, we performed a field-study at our industrial partner as a controlled statistical experiment (similar to a clinical study). We used in parallel an existing optimization method on a set of control branches and our size optimization based on the ISPO model on a set of test branches. From this study we derived that in a five-month period we could increase the mean relative realized revenue (mean of revenue divided by maximally possible revenue) by more than 1.5 percentage points (which means big money in economies of scale). To examine significance we applied Wilcoxon signed-rank test from statistics. We have not seen any published results that investigate the significance of practical results by this (or any other) statistical method, and we consider the introduction of controlled statistical experiments into the field of retail revenue management as a contribution in its own right.

1.3 Outline of the thesis

In Chapter 2 we outline the collaboration with our industrial partner as a historical process. We state different models and algorithms to decide for a supply policy in terms of lot-types. Moreover we show how the "newer" formulation SLDP with stochastic demand and monetary objective can be reduced to the "older" formulation LDP with deterministic demand and a non-monetary measurement. We present the model POP for price optimization which decides on weekly mark-downs and justify the decision for integrating price optimization already in the size optimization process.

All of these models refer to an empirical demand estimation method which was developed by involved colleagues. We outline the most important details in Chapter 3 and compare it with a – for our kind of data – common parametric approach.

Price optimization as part of DISPO is treated in detail in Chapter 4. For fixed supply we deduce dominance rules and apply them in a dynamic programming approach.

Before we present the stochastic program ISPO we outline some basics of stochastic programming in Chapter 5. We state the ISPO formally in Chapter 6.

In Chapter 7 we show how to reduce ISPO to the former model SLDP by fixing *price trajectories* – which are sequences of non-increasing prices from a set of predefined price steps – to scenarios. We present a theoretical idea for solving ISPO that is based on enumerating all possible combinations of assignments "scenario→price trajectory".

With the introduction of dual bounds for ISPO based on an extension of the wait-and-see solution from stochastic programming in Chapter 8 the idea now gets also practical interesting.

By combining enumeration of price trajectories from Chapter 6 with the idea to solve ISPO by enumerating SLDPs from Chapter 7 and the dual bounds from Chapter 8 we are able to state an exact Branch&Bound approach called ISPO-BAB for solving the ISPO to optimality. The approach is presented in Chapter 9. We apply ISPO-BAB to test instances with different settings. For practical use we propose our heuristic ISPO-PingPong. We perform ISPO-PingPong for different settings on a set of test data in terms of computation time and optimality gaps. Moreover, we compare ISPO-BAB with ISPO-PingPong on real instances.

With the use of ISPO-PingPong we performed a field-study at our industrial partner. We present results from preliminary studies and briefly go into some basics of statistical tests before we present the field study as a statistical experiment in Chapter 10.

We conclude this thesis with Chapter 11.

1.4 Preliminary remarks

1.4.1 Basics from mixed-integer linear programming

We apply some basics from mixed-integer linear programming MIP which are not explicitly treated in this thesis. For detailed informations about Branch&Bound, relaxations, LP relaxations, cover cuts, etc. we refer the reader to [Wol98], [Sch98] and [ANW06].

1.4.2 Labelling of own results

This thesis is a result of more than a five years long cooperation (2006 to 2011) at our industrial partner. We joined the project at 2009. Therefore not all topics in this thesis concerning the collaboration are a result of our own or our complete own work. We will use three terms to differentiate between us and the other colleagues:

- former DISPO-team: involved persons (in alphabetical order) were Konstantin Gaul, Tobias Kreisel, PD Dr. Sascha Kurz, Alexander Lawall, Prof. Dr. Jörg Rambau,
- DISPO-team: involved persons were Miriam Kießling, Tobias Kreisel, PD Dr. Sascha Kurz, Alexander Lawall, Prof. Dr. Jörg Rambau,
- we: Miriam Kießling.

1.4.3 Computational results

In this thesis we will state several computational results. If not otherwise specified they were provided by a machine with Intel(R) Xeon(R) processor with 2.33 GHz and 62 GB of RAM. We implemented all stated algorithms in C++. Whenever we will use a state-of-the-art solver for mixed-integer linear programs in our results we use IBM ILOG CPLEX, version 12.2 (as alternative also SCIP could be chosen; we tested the programs for SCIP in version 2.0.1 combined with SOPLEX-1.5.0 how it is included in the ZIBOPTSUITE-2.0.1).

Chapter 2

Collaboration with the industrial partner – historical progress

The Integrated Size and Price Optimization Problem ISPO is an enhancement of former models for the optimization of supply or size optimization that were developed during more than a five years long cooperation with our industrial partner. In this chapter we outline the main results related to the time before we developed DISPO. We show the historical progress from deterministic size optimization to integrated size and price optimization and outline the basic ideas of the implemented approaches. In Section 2.1 we treat the terms *lot-type* and *lot* in detail. The first model which assumes deterministic demand – the *Lot-type Design Problem* LDP presented by Gaul, Kurz and Rambau [GKR09], which is currently as a standard implemented at our industrial partner, is outlined at first in Section 2.2. We describe the so-called *SFA heuristic* which was introduced in [GKR10] as a solving method for the LDP. By additionally regarding stochastic demand and lot-opening costs we arrive at the SLDP – the *Stochastic Lot-type Design Problem* in Section 2.3. We show how to reduce the SLDP to the LDP which makes it possible to apply the SFA heuristic on it. Because both models do not contain all relevant properties of the sales process at our industrial partner as we outline in Section 2.4, we extend the SLDP by regarding the possibility of mark-downs, i.e. integrating price optimization. Price optimization as it was implemented by the former DISPO-team is treated in Section 2.5. We give a short outlook on integrated size and price optimization in Section 2.6.

2.1 Lot-types and lots

Our industrial partner already before the cooperation supplied its branches with lot-types. This is done to economize handling costs. A lot-type describes a prepackage which contains items of one product in different sizes and numbers. Mathematically we are given a lot-type by a n-tuple where n equals the number of sizes. The entries describe the number of items per size where we assume that the sizes are ordered increasingly. For example if we want to specify a prepackage containing 1 item of size S, 3 items of size M and 2 items of size L, we do this by the lot-type $(1, 3, 2)$. Before

the collaboration all branches were supplied by so-called standard prepackages. These are prepackages, i.e lot-types, which contain always just 1 item of the extreme sizes and 2 items for the middle sizes. An example would be the lot-type $(1, 2, 2, 2, 1)$.

A branch can be supplied by a specific number of prepackages, provided all prepackages are specified by the same lot-type. (To avoid handling costs it is not allowed to mix differing prepackages for supplying a branch.) For example supplying 2 times Lot-type $(1, 3, 2)$ means supplying 2 items of Size S, 6 items of Size M and 4 items of Size L.

2.2 The Lot-type Design Problem LDP

The Lot-type Design Problem LDP was the first formulation for optimization of supply at our industrial partner and was first presented in [GKR09].

2.2.1 Problem specification

We consider an article with a given *set of sizes* S. We want to supply each branch of a *set of branches* B with one lot-type from a *set of lot-types* L in a multiplicity from a *set of multiplicities* $M = \{1, \ldots, m^{\max}\}$. The lot-types are given by four parameters: the minimum supply per lot-type and size $v^{\min}$, the maximum supply per lot-type and size $v^{\max}$, the minimum supply per lot-type $vl^{\min}$ and the maximum supply per lot-type $vl^{\max}$. At the maximum κ different lot-types can be used for supplying the branches. The overall supply must lie in between a *lower bound* $\underline{I}$ and an *upper bound* $\bar{I}$. The supply shall meet the dependent demand $d_{b,s}, s \in S, b \in B$ for each branch b and size s as good as possible.

2.2.2 Problem formulation

The LDP is formulated as follows.

Problem 1 (LDP [GKR09])**.**

$$\min \sum_{b\in B}\sum_{\ell\in L}\sum_{m\in M} \mathrm{dist}^{\mathrm{LDP}}_{b,l,m} \cdot x_{b,l,m} \tag{2.1}$$

$$\text{subject to} \quad \sum_{\ell\in L}\sum_{m\in M} x_{b,l,m} = 1 \quad \forall b \in B, \tag{2.2}$$

$$\sum_{\ell\in L} y_\ell \le \kappa, \tag{2.3}$$

$$\sum_{m\in M} x_{b,l,m} \le y_l \quad \forall b \in B, \ell \in L, \tag{2.4}$$

$$I_{b,s} = \sum_{\ell\in L}\sum_{m\in M} m \cdot \ell_s \cdot x_{b,\ell,m} \quad \forall b \in B, s \in S, \tag{2.5}$$

$$I = \sum_{b\in B}\sum_{s\in S} I_{b,s}, \tag{2.6}$$

$$I \in [\underline{I}, \overline{I}], \tag{2.7}$$

$$x_{b,\ell,m} \in \{0, 1\} \quad \forall b \in B, \ell \in L, m \in M, \tag{2.8}$$

$$y_\ell \in \{0, 1\} \quad \forall \ell \in L. \tag{2.9}$$

Binary variables $x_{b,\ell,m}$ indicate if Lot-type ℓ is delivered to Branch b in Multiplicity m. If this is answered by "yes" the variable takes value one, otherwise zero. The binary variable y_ℓ takes value one if at least one branch is supplied by Lot-type ℓ, otherwise it takes value zero.

With Constraint (2.2) it is ensured that every branch is supplied by exactly one lot-type in one multiplicity. Constraint (2.4) connects the variables $x_{b,\ell,m}$ and y_l in such a way that y_ℓ can take value one only if lot-type ℓ is delivered to at least one branch. The adherence of the upper bound κ for the number of different lot-types is enforced by Constraint (2.3). With the constraints (2.5) and (2.6) it is ensured that the overall supply adheres to the lower and the upper bound; the variables $I_{b,s}$ here describe the supply for Branch b and Size s, I the overall supply and l_s the number of items of Size s in Lot-type ℓ.

The objective coefficients $\mathrm{dist}^{\mathrm{LDP}}_{b,l,m}$ measure the deviation between supply and demand. In our case we restrict ourselves to the L^1-Norm, which is also implemented at the industrial partner. For other measurements see [GKR09]. With a demand $d_{b,s}$ for Size s in Branch b, see Chapter 3 for the estimation method, the objective coefficients are given by

$$\mathrm{dist}^{\mathrm{LDP}}_{b,l,m} := \sum_{s\in S} |d_{b,s} - m \cdot l_s|. \tag{2.10}$$

Remark 1 (Lower and upper bounds for the overall supply [GKR09])**.** *For each product our partner first decides on an overall capacity D before the items are distributed to the particular branches. There are two reason why this amount is softened to the interval $\underline{I} \leq D \leq \overline{I}$ in Constraint (2.7):*

If for example the overall capacity D for an article was prime and there were two or more sizes for the considered product than the LDP would be infeasible. Such from a theoretical point of view the soft bound is needed to guarantee feasibility of the problem. The other reason is practical. Our partner does not always obtain the ordered supply from the supplier. As a rule there is a deviation between the ordered and the actual delivered amount. Deviations up to 5% from the ordered volume may occur.

Remark 2 (Complexity of the LDP [GKR09])**.** *The LDP is NP-hard. This is shown by reducing the p-median problem on it after restricting the set of multiplicities to the case $M = \{1\}$ and adapting the lower and upper bound in Constraint (2.7) in such a way that it is not a real restriction.*

Depending on the number of sizes, allowed lot-types and multiplicities the solving process for real-world instances by using state-of-the-art solvers for mixed-integer linear programs (MIPs) as SCIP or CPLEX can be very time-consuming (more than 5 hours) and therefore is not suitable for practical purposes. For that reason Gaul, Kurz and Rambau implemented the so-called *SFA heuristic*.

2.2.3 The Score-fix-adjust heuristic

In [GKR10] the Score-fix-adjust (SFA) heuristic for the LDP was proposed. The name of the heuristic stems from the three basic steps the heuristic consists of.

1. Score: The lot-types get scores in terms of how good they meet the demands of the branches.

2. Fix: For a given time period κ-subsets of the set of lot-types are traversed according to the scoring from the previous step. For each branch the best fitting lot-type from the considered subset and the related best fitting multiplicity is fixed.

3. Adjust: The multiplicities are adjusted to adhere to the bounds $\underline{I}$ and $\bar{I}$ for the overall supply.

In the Score-step each lot-type ℓ gets points according to the objective coefficient $\mathrm{dist}^{\mathrm{LDP}}_{b,l,m}$. This is done in the following way. For every branch b and lot-type ℓ first the best fitting multiplicity $m(b,\ell)$ is determined. This is the multiplicity $m \in M$ for which $\sum_{s\in S} |d_{b,s} - m \cdot l_s|$ is minimal. According to $\sum_{s\in S} |d_{b,s} - m(b,\ell) \cdot l_s|$ the lot-types $\ell \in L$ are ordered decreasingly. This yields an ordering of the lot-types in terms of how good they meet the demand of Branch b.

Starting from this for each branch the three locally best fitting lot-types can be determined. A score of 100 to the best fitting lot-type, a score of 10 to the second best fitting lot-type and a score of 1 to the third best fitting lot-type is added. (Of course this can be generalized to the first t best fitting lot-types and different scoring schemes.)

In the Fix-step the best κ-subsets of lot-types – best in terms of the highest sums of scores over all branches – are traversed. This is done for a predefined time period trusting that the most promising selections of lot-types were checked. For the considered κ-subset $L' \subseteq L$ for a branch b the lot-type $\ell' \in L'$ is fixed which minimizes $\sum_{s\in S} |d_{b,s} - m(b,\ell) \cdot l'_s|$. Such, a preliminary supply policy with a corresponding overall supply I' is specified. If $I' \in [\underline{I}, \bar{I}]$ the supply policy is valid. Otherwise the Adjust-step, see below, has to be performed to establish feasibility. If the supply policy yields a smaller objective value of the LDP than the already considered ones or if it is the first considered one, we update our best found solution correspondingly.

The Adjust-step assures the adherence of the bounds $\underline{I}$ and $\bar{I}$ for the overall supply. Fixing the best fitting lot-type from the considered κ-subset with best matching multiplicity for each branch in the previous step might violate Constraint (2.7). There are two cases of infeasibility:

1. $I' < \underline{I}$

2. $I' > \bar{I}$

In the first case supply is increased until the lower bound is met. This is done in a greedy way: The branch for which increasing the currently fixed multiplicity by one is valid and leads to the smallest additional costs in terms of the objective function is determined. The multiplicity is increased by one and fixed. This procedure is iterated unless the lower bound for the overall supply $\underline{I}$ is met.

The proceeding in the second case is similarly. Supply iteratively is reduced until the upper bound is met. From all branches that are at least supplied with a lot-type in multiplicity 2 we choose the branch for which decreasing the multiplicity by one leads to the smallest additional costs.

For 36 real instances the authors performed the SFA heuristics with a computation time of one second. This led to a mean optimality gap of 0.327% while the highest gap amounts to 2.114%.

2.2.4 Implementation at the industrial partner

Gaul, Kurz and Rambau [GKR10] performed a preliminary study at our industrial partner to evaluate if the LDP performs better than manual planning of supply. Previously,

the branches were all supplied by the same standard lot-type as we described in Section 2.1.

In [KRSW08] the authors could show that the size dependent demand among different sizes at our industrial partner actually varies and that the LDP together with their presented demand estimation method, see Chapter 3, could increase the gross yield about 0.85 percentage points.

Since 2006 the LDP, more precisely, the SFA heuristic is implemented at the partner and used for nearly all fashion articles which are supplied in terms of lots.

2.3 The Stochastic Lot-type Design Problem SLDP

The former DISPO-team enhanced the LDP to the Stochastic Lot-type Design Problem, the SLDP. The SLDP can be understood as an intermediate model between the deterministic LDP and the final stochastic model ISPO integrating price optimization. In the SLDP different scenarios with scenario probabilities and scenario dependent demands estimated from historical data are treated. While in the LDP abstract costs in form of the L_1-norm are considered and over- and undersupply are treated the same, now monetary asymmetric costs are imposed for the deviation between supply and demand. With these costs the SLDP can be seen as a first step in integrating the sales process in the size optimization. Monetary measurement now allows also to take other costs into account. On the one hand pick costs which arise from arranging the lot-types to lots and on the other hand lot-opening costs which arise from the fact that each additional supplied lot-type leads to higher logistic effort.

2.3.1 Problem specification

We consider an article with a given set S of *sizes*. We want to deliver each branch from a set B of *branches* with one lot-type from a set L of *lot-types* in a multiplicity from the set $M = \{1, \ldots, m^{\max}\}$ of *multiplicities*. At the maximum κ different lot-types are allowed to use for supply. For the ith supplied new lot-type from the set L *lot-opening cost* δ_i arise . For every handgrip needed for putting together the lot-types to lots *pick cost* pcost arise. The overall supply must lie in between a *lower bound* $\underline{I}$ and an *upper bound* $\bar{I}$. Now we consider a set E of different *scenarios* with *scenario probabilites* $\mathrm{Prob}(e), \forall e \in E$. With given demands $d^e_{b,s}$ for each size s, branch b and scenario e an oversupply is penalized by *acquisiton price* ap minus *salvage value* $\pi_{p_{\max}}$, an undersupply by *starting price* π_0 minus ap. This means that it is assumed, that each undersupply would lead to a loss of the full starting price while each oversupplied item can just be sold for the salvage value. The aim is to minimize the expected overall costs, i.e. the sum of the handling costs – lot-opening and pick cost – together with the expected costs for oversupply and undersupply. In terms of demand estimation and the estimation of the probabilities $\mathrm{Prob}(e), e \in E$ see Chapter 3.

2.3.2 Modelling the SLDP

Before we introduce the entire model we first focus on the coefficients in the objective: The expected dependent demand $d_{b,s}$ for Branch b and Size s is given by $d_{b,s} = \sum_{e \in E} \mathrm{Prob}(e) \cdot d^e_{b,s}$, where $d_{b,s}$ equals the dependent demand in the LDP. Now asymmetric costs for over- and undersupply are introduced. An oversupply is penalized by acquisition price minus salvage value $\mathrm{ap} - \pi_{p_{\max}}$, an undersupply by starting

price minus acquisition price $\pi_0 - \mathrm{ap}$. Then the cost arising by suppling Branch b with Lot-type ℓ in Multiplicity m are given by $\mathrm{dist}^{\mathrm{SLDP}}_{b,\ell,m}$ which is defined as

$$\mathrm{dist}^{\mathrm{SLDP}}_{b,\ell,m} := \sum_{s\in S}(\max\{m\cdot l_s - d_{b,s}, 0\}\cdot(\mathrm{ap}-\pi_{p_{\max}})+\max\{d_{b,s}-m\cdot l_s, 0\}\cdot(\pi_0-\mathrm{ap})). \tag{2.11}$$

The Stochastic Lot-type Design Problem SLDP is modeled as follows:

Problem 2 (SLDP)**.**

$$\min \sum_{b\in B}\sum_{\ell\in L}\sum_{m\in M}\left(\mathrm{dist}^{\mathrm{SLDP}}_{b,\ell,m} + m\cdot \mathrm{pcost}\right)x_{b,\ell,m} + \sum_{i=1}^{\kappa}\delta_i\cdot z_i \tag{2.12}$$

$$\text{subject to} \quad \sum_{\ell\in L}\sum_{m\in M} x_{b,\ell,m} = 1 \quad \forall b\in B, \tag{2.13}$$

$$\sum_{m\in M} x_{b,\ell,m} \le y_\ell \quad \forall b\in B, \ell\in L, \tag{2.14}$$

$$\sum_{\ell\in L} y_\ell \le \sum_{i=1}^{\kappa} z_i, \tag{2.15}$$

$$z_i \le z_{i-1} \quad i = 1\ldots,\kappa, \tag{2.16}$$

$$I_{b,s} = \sum_{\ell\in L}\sum_{m\in M} m\cdot\ell_s\cdot x_{b,\ell,m} \quad \forall b\in B, s\in S, \tag{2.17}$$

$$I = \sum_{b\in B}\sum_{s\in S} I_{b,s}, \tag{2.18}$$

$$I \in [\underline{I}, \overline{I}], \tag{2.19}$$

$$x_{b,\ell,m} \in \{0,1\} \quad \forall b\in B, \ell\in L, m\in M, \tag{2.20}$$

$$y_\ell \in \{0,1\} \quad \forall \ell\in L, \tag{2.21}$$

$$z_i \in \{0,1\} \quad i = 1,\ldots,\kappa. \tag{2.22}$$

Most constraints are similar to the ones of the LDP. At this point we explain only the differences and refer the reader to Problem 1.

To take handling costs into account we introduce the binary variables $z_i, i = 1\ldots,\kappa$ which indicate if at least i different lot-types are opened. Constraint (2.15) links the variables z_i with y_l. By Constraint (2.16) it is ensured that z_i can take value one only if z_{i-1} also does. The additional costs for opening new lot-types are added in the objective function and for every delivered lot pick costs $m\cdot\mathrm{pcost}$ arise in addition to the costs for over- and undersupply (2.12).

Corollary 1 (Complexity of the SLDP)**.** *The SLDP is NP-hard.*

Proof. If we set δ_i to zero for $i = 1,\ldots,\kappa$ in the SLDP, we obtain an LDP with changed objective coefficients because the constraints (2.15) and (2.16) in this case are equivalent to Constraint (2.3). That means, we can reduce the LDP in polynomial time to the SLDP. Because the LDP is NP-hard – as stated in Remark 2 – the SLDP is, too. □

2.3.3 Solving the SLDP by the LDP

The SLDP simplifies to an LDP if we set all lot-opening costs to zero. We now show that in similar way we are able to determine the optimal solution of the SLDP as the best solution resulting from solving κ LDPs.

For $i = 1, \ldots, \kappa$ we consider the following formulation of the LDP.

Problem 3 (LDP(i))**.**

$$\min \sum_{b\in B}\sum_{\ell\in L}\sum_{m\in M}\big(\mathrm{dist}^{\mathrm{SLDP}}_{b,\ell,m} + m\cdot \mathrm{pcost}\big)x_{b,\ell,m} \tag{2.23}$$

$$\text{subject to} \quad \sum_{\ell\in L}\sum_{m\in M} x_{b,l,m} = 1 \quad \forall b\in B, \tag{2.24}$$

$$\sum_{\ell\in L} y_\ell \le i, \tag{2.25}$$

$$\sum_{m\in M} x_{b,l,m} \le y_l \quad \forall b\in B, \ell\in L, \tag{2.26}$$

$$I_{b,s} = \sum_{\ell\in L}\sum_{m\in M} m\cdot \ell_s \cdot x_{b,\ell,m} \quad \forall b\in B, s\in S, \tag{2.27}$$

$$I = \sum_{b\in B}\sum_{s\in S} I_{b,s}, \tag{2.28}$$

$$I \in [\underline{I}, \overline{I}], \tag{2.29}$$

$$x_{b,\ell,m} \in \{0,1\} \quad \forall b\in B, \ell\in L, m\in M, \tag{2.30}$$

$$y_\ell \in \{0,1\} \quad \forall \ell\in L. \tag{2.31}$$

The LDP(i) is an LDP with the restriction that at most i instead of κ different lot-types are allowed for supply, Constraint (2.25). The coefficients of the variables $x_{b,\ell,m}$ in the objective function are these from the SLDP. Opening-costs for new lot-types are not regarded.

Having solved the LPD(i) with the optimal solution $(x^*(i), y^*(i))$ we can compute the corresponding overall opening costs by adding

$$\sum_{j=1}^{\sum_{\ell\in L} y^*_\ell(i)} \delta_j$$

to the optimal objective value $z^*_{\mathrm{LDP}(i)}$. Thus, by solving the LDP(i) for each $1 \le i \le \kappa$ separately we obtain the optimal supply for each possible allowed number of different lot-types. Adding the opening costs to the related objective value yields the optimal objective value of the SLDP. The LDP(i) for which the objective value plus the corresponding opening costs is minimal among $1 \le i \le \kappa$ then yields the optimal solution of the SLDP.

Theorem 1 (Deducing the optimal solution of the SLDP from the LDP)**.** *With $z^*_{\mathrm{LDP}(i)}$ we denote the optimal objective value of the* LDP(i)*. The corresponding optimal solutions are denoted by $x^*(i)$ and $y^*(i)$. With z^*_{SLDP} we denote the optimal objective value of the* SLDP *and with x^*, y^* and z^* the related values of the variables. We define*

$$i^* := \arg\min_{i=1,\ldots,\kappa} \left\{ z^*_{LDP(i)} + \sum_{j=1}^{\sum_{\ell\in L} y^*_\ell(i)} \delta_j \right\}. \tag{2.32}$$

Then the optimal objective function value of the SLDP *is given by*

$$z^*_{\text{SLDP}} = z^*_{\text{LDP}(i^*)} + \sum_{j=1}^{\sum_{\ell\in L} y^*_\ell(i^*)} \delta_j. \tag{2.33}$$

It is $x^* = x^*(i^*)$, $y^* = y^*(i^*)$ *and* $z^*_j = 1$ *for* $j = 1, \ldots, \sum_{\ell\in L} y^*_\ell(i^*)$ *and* $z^*_j = 0$ *for* $\sum_{\ell\in L} y^*_\ell(i^*) < j \leq \kappa$.

Proof. It is i^+ the number of used lot-type according to the optimal solution of the SLDP. If we would set $\kappa = i^+$ in the SLDP this would yield the same optimal solution. We call the SLDP restricted to maximal $\kappa = i^+$ different lot-types SLDP(i^+). The corresponding optimal objective value is denoted by $z^*_{\text{SLDP}(i^+)}$. The LDP(i^+) yields a supply $x^*(i^+)$ that minimizes $\sum_{b\in B}\sum_{\ell\in L}\sum_{m\in M}\left(\text{dist}^{\text{SLDP}}_{b,\ell,m} + m\cdot\text{pcost}\right) x^*(i^+)_{b,\ell,m}$ for maximal i^+ different lot-types not regarding lot-opening costs. Because the z_i and the lot-opening costs are independent from the selected lot-types and depend only on the number of them the LDP(i^+) yields the same optimal solutions in terms of the supply than the SLDP(i^+). To obtain the same solution we could set the lot-opening costs δ_i in the SLDP(i^+) to zero, compute the optimal supply and later on add the costs δ_i for the i^+ used lot-types. This is the same as solving the LDP(i^+) and adding the corresponding lot-opening costs. Overall that means

$$z^*_{\text{SLDP}} = z^*_{\text{SLDP}(i^+)} = z^*_{\text{LDP}(i^+)} + \sum_{j=1}^{\sum_{\ell\in L} y^*_l(i^+)} \delta_j.$$

It is $z^*_{\text{SLDP}} \geq z^*_{\text{LDP}}(i) + \sum_{j=1}^{\sum_{\ell\in L} y_\ell(i^*)} \delta_j$ for all $i = 1, \ldots, \kappa, i \neq i^+$. Otherwise the SLDP would yield an optimal solution with less or more than i^+ lot-types.

By setting $i^* = i^+$ the claim follows.

□

Remark 3. *In order to compute the optimal solution of the* SLDP *we propose to solve the* LDP(i)*s in ordering* $i = \kappa, \ldots, 1$. *If the* LDP(i) *yielded a supply policy with just* $i^- < i$ *lot-types we would not have to solve the* LDP(j) *for* $i^- \leq j < i$. *The* LDP(j)*s would yield the same optimal solution as the* LDP(i). *So traversing the* LDP(i)*s in order* $i = \kappa, \ldots, 1$ *may reduce the computational effort.*

By reducing the SLDP to the LDP now it is possible to apply solving methods for the LDP – as the described SFA heuristic – to the SLDP. Later on, in Chapter 9, we will mention how this property can be exploited when solving the Integrated Size and Price Optimization Problem ISPO which is discussed in Chapter 6.

2.3.4 A column generation approach

In [KKR11a] an exact column generation approach for the LDP is presented by the DISPO-team. The approach is guided by two main ideas[1].

- Considering the restricted master problem (RMP) with only a subset $L' \subset L$ of lot-types

[1] For further information about column generation we refer the reader to [LD05],[LD11] or [Lüb10]

- Solving the LDP for the *most promising* subset of lot-types $\bar{L} \subseteq L'$ exactly

We will sketch the main parts of the approach. For further details we refer the reader to [KKR11a].

A restricted master problem RMP of the LP relaxation of the LDP is considered. The only difference to the LP relaxation is that only a subset of the lot-types are considered. Thus, because the optimal solution may not be contained, the restricted master problem yields an upper bound for the LP relaxation of the original problem.

1. At first, a starting solution (x^*, y^*) of the LDP is determined. This can be done via an adapted version of the SFA heuristic. The used lot-types, i.e. lot-types with $y^*_\ell = 1$ then are added to the initial subset L' of lot-types. Additionally the three best fitting lot-types for each branch – as they result from the score-step of the SFA-heuristic, see 2.2.3 – are added to L'. The lot-types from the set L' are the only lot-types that are considered in the RMP at the beginning.

2. With $(x^{\text{RMP}}, y^{\text{RMP}})$ we denote the optimal solution of the RMP. The set of most promising lot-types $\bar{L}$ is the subset of all lot-types from the set L' with $y^{\text{RMP}}_l \geq \varepsilon$ where ε is a small constant, for example $\varepsilon = 0.15$. If the optimal objective value of the RMP is smaller than the objective value of the current best integer solution (x^*, y^*) the LDP restricted to $\bar{L}$ is solved exactly and possibly the currently best integer solution (x^*, y^*) is updated. (If the RMP yields an optimal value higher than the to (x^*, y^*) corresponding objective value the set L' cannot contain the optimal subset of lot-types. Because the RMP is a relaxation of the LDP that contains only the lot-types L' the optimal objective value is a lower bound for the LDP restricted to the set L' of lot-types. Such, we are not able to obtain a better integer solution than (x^*, y^*) by only regarding the lot-types from the set L'.) Cover cuts are added to the RMP to forbid that the optimal solution of the RMP yields $\bar{L}$ as the set of most promising lot-types again. This implies a branching on the set $\bar{L}$ and the rest of lot-types L' currently considered in the RMP.

3. Whenever the optimal function value of the RMP is higher than or equals the objective value of the current best solution (x^*, y^*) the pricing step is performed in which – if possible – new lot-types are added to the RMP – i.e. L' is updated and the RMP is solved again and so on. Whenever the optimal objective function value of the RMP is smaller than the to (x^*, y^*) related objective function value of the LDP, then we update the subset of most promising lot-types L' and branch on this subset, i.e. perform Step 2. If the optimal objective function value of the RMP exceeds or equals the objective value of the LDP corresponding to (x^*, y^*) and no more lot-types are/can be added to the RMP than we end up at this point and return (x^*, y^*) as optimal solution.

The results in [GKR09] show that for real-world instances the maximum amount of time for solving can be reduced from 36 minutes to 4 seconds. Even very large instances – for which state-of-the-art MIP solvers fail – can be solved in less than 16 minutes.

2.4 Reasons for integrating price optimization

The introduction of monetary costs in Problem 2 is a first step in integrating the sales process in the size optimization. But by penalizing oversupply with acquisition price

minus salvage value we assume that each oversupplied item cannot be sold during the regular selling time and only during sellout. The penalization of undersupply implicitly assumes that if an additional item of the related size to the related branch had been supplied, it could have been sold for the full starting price.

But this does not describe the situation at our industrial partner correctly: If an article is a bad seller prices are marked down during the sales process hoping that more than the salvage value can be earned for the left over items. Therefore the formulation of the costs accrued by oversupply are very vague. If our partner performs a mark-down for a product it happens in all branches and sizes simultaneously in the same way. Thus, particular branches and sizes in which the article is possibly a good seller are not explicitly regarded. What counts is to maximize the overall revenue over all branches and sizes. Therefore an undersupply for one particular branch and size would not always lead to a loss of the full sales price as it is assumed in the formulation of the objective of the SLDP.

Hence, the conclusion is to integrate the possibility of mark-downs during the sales process already in the decision on the supply policy.

2.5 Price optimization

The former DISPO-team performed price optimization on its own before we linked it to the size optimization process. We specify the problem in Subsection 2.5.1 and state the mixed-integer linear program POP in Subsection 4.1.1. Specialties regarding this formulation are treated in Subsection 2.5.3. In Subsection 2.5.4 we go into the practical application of price optimization – price optimization with receding horizon, POP-RH.

2.5.1 Problem specification

An instance of the Price Optimization Problem POP consists of a set $P = \{0, \dots, p_{\max}\}$ of *price indices* with related decreasing positive *prices* $\pi_p, p \in P$. The price π_0 is the *starting price*, $\pi_{p_{\max}}$ the *salvage value*.

Moreover a set $K = \{0, \dots, k_{\max}\}$ of *sales periods* is given with $k_{\max}$ being the *sellout period*. We call the periods $k = 0, \dots, k_{\max} - 1$ the *real sales periods*. A value $\rho > 0$ for *discounting* during the sales process is given. For the first k_{obs} *observation periods* no mark-downs are allowed. In the *sellout period* $k_{\max}$ all remaining items are sold for the salvage value.

We consider a set B of *branches* and a set S of *sizes* with an initial stock $I_{b,s}$ for each size s in each branch b.

For every *scenario* e from a *set of scenarios* E the *dependent demand* for each size s in each branch b for each price to the related price index $p < p_{\max}$ in each real sales period k is given by $d^e_{k,p,b,s}$.

Our task is to allocate a price to each period so that the prices are non-increasing during the sales process and the expected revenue over the scenarios is maximized.

2.5.2 Problem formulation

The former DISPO-team stated the Price Optimization Problem as follows.

Problem 4 (The Price Optimization Problem (POP))**.**

$$\max \mathbb{E}_{e\in E} \sum_{k\in K} \exp(-\rho k)\Big(\sum_{b\in B}\sum_{s\in S} r^e_{k,b,s}\Big) \tag{2.34}$$

$$\sum_{p\in P} u^e_{k,p} = 1 \quad \forall k \in K, e \in E, \tag{2.35}$$

$$u^e_{k,0} = 1 \quad \forall k \in K : k < k_{obs}, e \in E, \tag{2.36}$$

$$u^e_{k_{\max},p_{\max}} = 1 \quad \forall e \in E, \tag{2.37}$$

$$u^e_{k-1,p_1} + u^e_{k,p_2} \leq 1 \quad \forall k \in K : k > 0, p_1, p_2 \in P : p_2 < p_1, e \in E, \tag{2.38}$$

$$v^e_{0,b,s} = I_{b,s} \quad \forall b \in B, s \in S, e \in E, \tag{2.39}$$

$$v^e_{k-1,b,s} - v^e_{k,b,s} = \sum_{p\in P} w^e_{k-1,b,s,p} \quad \forall k \in K : k > 0, b \in B, s \in S, e \in E, \tag{2.40}$$

$$\sum_{p\in P} w^e_{k,b,s,p} \leq v^e_{k,b,s} \quad \forall k \in K, b \in B, s \in S, e \in E, \tag{2.41}$$

$$w^e_{k,b,s,p} \leq u^e_{k,p} \cdot d^{\hat{e}}_{k,p,b,s} \quad \forall k \in K : k < k_{\max}, b \in B, s \in S, p \in P : p < p_{\max}, e \in E, \tag{2.42}$$

$$w^e_{k_{\max},b,s,p_{\max}} = v^e_{k_{\max},b,s} \quad \forall b \in B, s \in S, e \in E, \tag{2.43}$$

$$r^e_{k,b,s} = \sum_{p\in P} \pi_p \cdot w^e_{k,b,s,p} \quad \forall k \in K, b \in B, s \in S, e \in E, \tag{2.44}$$

$$u^e_{k,p} \in \{0,1\} \quad \forall k \in K, p \in P, e \in E, \tag{2.45}$$

$$w^e_{k,b,s,p} \geq 0 \quad \forall k \in K, b \in B, s \in S, p \in P, e \in E, \tag{2.46}$$

$$v^e_{k,b,s} \geq 0 \quad \forall k \in K, b \in B, s \in S, e \in E, \tag{2.47}$$

$$r^e_{k,b,s} \geq 0 \quad \forall k \in K, b \in B, s \in S, e \in E. \tag{2.48}$$

The binary variable $u^e_{k,p}$ indicates if the to Price index p related price is allocated to Period k for Scenario e. If this is the case the variable takes value one, otherwise value zero. Equation (2.35) enforces the assignment of exactly one price/price index p to each period. For a given number of periods k_{obs} from the beginning of the sales process the starting price is enforced by Equation (2.36). In the last period – the sell-out – the salvage value is fixed by Equation (2.37). We forbid increasing prices by Equation (2.38). The following restrictions model the dynamics of the sales process using some dependent variables. The fractional variable $v^e_{k,b,s}$ specifies the scenario dependent stock level in Period k in Branch b for Size s. The initial stock per branch b and size s is given by the corresponding supply $I_{b,s}$ – which in this case is a parameter (2.39). The fractional variable $w^{\hat{e}}_{k,b,s,p}$ measures the scenario dependent sales in Period k in Branch b for Size s for the price with index p. Equation (2.40) describes the change of stock levels from one period to another. Inequality (2.41) models that there can be no more sales than stock, and in Inequality (2.42) we require that, only if the price with Price index p is chosen, there can be sales at the Price index p of at

most the demand at Price index p. Because the objective favors larger sales, the sales variables at a price in an optimal solution will become exactly the minimum of stock and demand at that price. Finally, we capture by Equation (2.44) the yield in terms of money. Only the independent price assignment variables need to be binary (2.45). The dependent variables capturing the dynamics of stocks, sales, and yields are required to be nonnegative in (2.46) through (2.48). Our objective is to maximize the expected revenue.

2.5.3 Justificating the problem formulation

In the problem formulation of the POP fractional sales and stocks are considered. In the most cases for an article we can observe no sales per week, branch and size.[2] In consequence of this our demand estimation – see Chapter 3 – yields mean demands that mostly lie in between zero and one. Therefore in POP the sales and stock variables also describe fractional mean sales and stocks.

In POP a set of scenarios E is considered. A scenario from the set E refers to the whole selling time for the considered product and does not change over time. The former DISPO-team considered the scenarios

$$E = \{\text{“low seller”}, \text{“normal seller”}, \text{“high seller”}\}.$$

One could think about formulating price optimization as a so-called *multi-stage* problem where the scenario changes over time according to the observed sales. One could also think about including more than three scenarios. By a finer partitioning of the set of scenarios the problem formulation of the POP would get much more complicated. Moreover our data basis for estimation – transaction data of one commodity group for a time frame of about a year – is too small to consider larger sets of scenarios. The samples for the particular scenarios would be to small for a reasonable estimation. Later on, in Chapter 10 we will see that demand estimation for ISPO is very vague. There are high deviations among products from the same sample. We can not make a point of an potential improvement of considering more recourse stages or more scenarios with only inaccurate estimations of demand. It is questionable if the potential improvement would justify a complication of the model.

As an alternative to a multi-stage formulation and to profit from current sales figures a different proceeding presented in the next subsection was applied.

2.5.4 Price optimization with receding horizon – POP-RH

Price optimization is performed similarly to a *closed-loop* process with a receding horizon. After an observation time of k_{obs} periods, the price optimization process starts with the knowledge about the scenario in effect deduced from current sales figures. The scenario in terms of the sales figures is updated after each period, see Chapter 3, 3.2.6 for details, and in Sales period k the Price Optimization Problem for the set $K = \{k+1, \ldots, k_{\max}\}$ of periods, and E containing only the scenario in effect, is solved again. This proceeding is similar to *model predictive control* as a special field of dynamic programming, see for example [Ber05] or [GPM89]. In the context of smaller field studies the former DISPO-team solved the particular models per period (week in the case of application) by state-of-the-art MIP solvers. In Chapter 4 we will see that the typical computation time for a standard selling time of 13 periods with CPLEX

[2] see for example Table 3.4.1 in Chapter 3

amounts more than 8 hours. This is too slow for practical use where for more than 4000 products within two days about mark-downs has to be decided.

2.6 Integrated size and price optimization

We adopt the proceeding of the former DISPO-team and prefer a so-called *two-stage* instead of a *multi-stage* model for integrated size and price optimization. In the first stage we decide on a supply and in the second stage according to the scenario in effect price optimization is performed.

An important aspect – also for the already presented models – is the estimation of demand. We use an empirical method which was developed by the former DISPO-team. It is topic of the next chapter.

To benefit from all current information we apply price optimization in the same way as described above after we determined the optimal supply with our integrated model. Thus, our decision support system for integrated size and price optimization DISPO consists of two parts. One is to solve ISPO one time and supply the branches according to the solution. The other one is to perform price optimization with receding horizon POP-RH every sales period again. Therefore, we have to develop faster approaches for price optimization. In Chapter 4 we treat price optimization as it takes part in DISPO in detail and present a dynamic programming approach for fixed supply before we present the Integrated Price and Size Optimization Problem (ISPO) in Chapter 6. The remainder of the thesis is devoted to the exact and heuristic solution of ISPO and real-world experiments in terms of DISPO.

Chapter 3

Demand estimation

An important aspect of DISPO is the estimation of the dependent demand. There are some special features about the situation at our industrial partner which have to be regarded. The considered fashion products are only sold once and are never offered again. Thus, historic sales data can only be used on a higher aggregation level, e.g., the average historic demand at a price in a sales week on the commodity group level. The number of sales also depends from the popularity of the observed products. The demand estimation method should handle this aspect.

Since the supplies per branch and size of a single product are zero, one, or two in most cases, we can expect that historic sales data will only give us very coarse information. Only sales can be observed and not the real demand. If a size is sold out in a branch we do not know if the actual demand for this size in this branch was higher than the observed sales. This kind of data is also called *right-censored*. The additional sales that would have occurred if the corresponding supply had been higher in literature are known as *lost sales*. References can be found in Section 3.1. Here we also state some estimation approaches from literature and outline the applicability on our situation. We present an empirical estimation approach for DISPO developed by the former DISPO-team in Section 3.2. A parametric approach, logistic regression together with maximum likelihood estimation is outlined in Section 3.3. We present a logistic regression model for demand estimation for DISPO in Section 3.4 and compare it to the empirical estimation method in Section 3.5.

3.1 Literature review

The most common approach for estimating dependencies of a dependent variable from one or more independent variables is linear regression, see for example [Har10].

According to Breen [Bre96] there are two different procedures to handle with censored data. One is to omit all data which might be censored and then to apply linear regression. The other one is to estimate the censored data. A common approach to do this for metric data is the so-called *tobit model*, invented by James Tobin [Tob58], where it is assumed that the censoring-point for each observation is equal. Tobit regression is a special case of the so-called censored regression. Here each observation might be censored at a different point. The models are commonly estimated via *maximum likelihood estimation*.

In [VvRR12] Vulcano et al. use the so-called *expectation-maximization method*

(EM method) to estimate the demand for a product with right-censored sales data. The EM method in general was first proposed by Dempster et al. [DLR77] to estimate dependencies on the basis of incomplete observations. The basic idea of the EM-algorithm is the following: By starting with an initial model alternately the missing observations according to the given data and the current model parameters are estimated and the parameters are adapted (maximization) via maximum likelihood estimation. Vulcano et al. in the expectation step estimate *primary (or first choice) demand*. This is the actual demand, while the sales are only incomplete observations of primary demand. The primary demand is estimated by assuming that the arrivals of the customers are Poisson-distributed over time. If a customer wants to buy a product there are two possibilities: Either the product is available then he will buy it or it is not. If the product is not available he might buy a similar product either at the same provider at a competitor. This happens according to a so-called market share which is given as input of the algorithm.

In the maximization step the estimated primary demand is used to estimate the parameters in terms of preference values for the observed products via maximum likelihood estimation.

A similar approach also using the EM-algorithm with Poisson-distributed arrivals is outlined in [ADG98].

Huh et al. [HLRO11] applied the so-called Kaplan-Meier Estimator, see for example [ABG10], to stochastic inventory control problems with censored demand to decide on the time items have to be reordered. The *Kaplan-Meier estimator* – also *product limit estimator* – is an approach from survival analysis. The non-parametric approach examines how long the observed individuals stay in a state.

It is often applied in medial resarch where the question is: "Does the patient survive or will he die?" The probability $S(t)$ that a member has a life time exceeding t is considered. For a sample with size N the observed times t_i until death are ordered increasingly. The probability of surviving t_i days can be seen as the probability of surviving day t_i after living t_{i-1} days multiplied by the chance of surviving t_{i-1} days. Thus, $S(t)$ is given by $S(t) := \prod_{t_i<t} \frac{n_i-d_i}{n_i}$ where d_i is the number of deaths at time i and n_i the number of survivors at the end of t_{i-1}.

A special estimation method for binary response is *logistic regression*. While linear regression might lead to outcomes that can be negative or higher than one, logistic regression measures the probability for outcome one or zero. The dependent variables are transformed to the so-called *odds*. The odds describes the relation of the probability that the event happens to the probability that it does not – the odds can only take values higher than zero. By taking the logarithm the odds is transformed to the so-called *logit*. As a linear function of the independent variables with a range between $-\infty$ and $+\infty$ it is commonly estimated by *maximum likelihood estimation*.

A similar model is the so-called *probit model*, see for example [Rya08]. It distinguishes from logistic regression by the assumption is that the error is normally distributed. For the logistic model we assume a distribution according to the so-called *logistic function*. Both models belong to the class of the so-called *generalized linear models*. In contrast to ordinary linear regression models generalized models do not assume that the residuals are normally distributed. Both can be extended for ordinal data. For logistic regression we will outline this later on. For further information about generalized linear models and maximum likelihood estimation see for example [Har10] or [Rya08].

We did not find any general comparison of logistic and probit regression in literature.

Applicability

To omit right-censored data in our case is not possible. Because the most sizes per branch are supplied by maximal one or two items and nearly all observations are right-censored this would lead to a tiny sample which could not serve as basis for demand estimation whatsoever. Thus, we have to deal with censored observations.

For our right-censored and ordinal data linear regression as mentioned above is not suitable. Ordinary linear regression may lead to negative values for the dependent variable, i.e. in our case the demand. Because the supply and consequently the censoring-point for each size and branch differs the tobit model is not an alternative, rather a general censored regression model. But also this is not conceived for integer numbers of outcomes.

The EM approach by Vulcano et al. described above in principle appears promising. However, the approach needs a set of comparable products as input. One could consider all items of the same commodity group or sub commodity group as comparable but this is not the case. The articles differ in color, fashion, etc. and most important price. But even if our industrial partner could commit us lists of comparable products there would be another difficulty: Because there is no reordering of products and the products have different sales starts we have to regard that the list of comparable articles may change over time. Moreover, incomparability caused by mark-downs would have to be regarded.

The Kaplan-Meier approach as applied in [HLRO11] can not directly be adopted. In our situation the items are not reordered. So far, we did not see how the method could be adapted to our situation where we also have to include the possibility of mark-downs during the selling time.

From all mentioned methods from literature the for us most promising approach is the ordinal logistic regression model. This can explicitly deal with small integer outcomes. Moreover we can include price dependencies and dependencies in terms of the popularity of the observed product. By including the current stock as independent variable we estimate sales not demand and have not to deal with right-censored observations.

3.2 Empirical estimation

All models and results in this thesis are based on an empirical demand estimation developed by the former DISPO-team. Parts of this method were already implemented to estimate the demand in terms of the LDP the SLDP and the Price Optimization Problem POP.

Because there are no publications including a detailed description we outline the method at this point.

The overall demand D is considered as an exogenous quantity.[1]

The estimator circumvents the difficulties arising from varying popularity among different articles by considering the conditional probability that – if an item is sold – this happens in Branch b and Size s. The conditional probability is also called the *mean amount of demand* $\delta_{b,s}$ for Branch b and Size s.

Additionally we consider the mean amount of demand δ_b for Branch b. It is the conditional probability that if an item is sold this happens in Branch b.

[1] This coincides with the proceeding at our industrial partner. Before the supply according to lots takes place, the purchaser decides on the overall number D of supplied items.

With these quantities the fractional *mean demand* $d_{b,s}$ per branch b and size s for the sets B of branches and S of sizes is computed. We outline the approach in Subsection 3.2.1. We show how to include the set E of scenarios in Subsection 3.2.2 before we split up $d_{b,s}$ to the sales periods – in our case weeks – $K \setminus \{k_{\max}\}$ and include price dependency for the price indices $P \setminus \{p_{\max}\}$ in the subsections 3.2.3 and 3.2.4. In Subsection 3.2.5 we combine the single estimates to arrive at the estimate we use for DISPO. Concluding, in Subsection 3.2.6, we describe how to adapt the demand estimation according to a scenario in effect, as it is the case when we perform price optimization with receding horizon.

Demand is always estimated by pooling observations from products from the same commodity group. A smaller division for example to sub commodity groups in our case would lead to an insufficient amount of data. Commodity groups are for example "women overgarments classic" or "women overgarments fashion" or "men trousers".

3.2.1 Relative demand estimation

To exclude the influence of the popularity of the different articles and also to reduce the influence of lost-sales we consider sales just until the day when 50% of the observed product's overall supply (over all branches and sizes) is sold. The advantage of doing so is that we obtain a measurement of the sales speed for different sizes and branches. If we considered the complete selling time per article varying behavior among branches and sizes – because due to mark-downs nearly all items would be sold out – would not longer be recognizable.

For article $a \in \mathcal{A}$ we denote the amount of sales (over all branches and sizes) until the point in time when 50% of the overall supply for a are sold by $sal50^a$. The amount of sales for Branch b for the same time frame is denoted by $sal50^a_b$ and for Branch b and Size s by $sal50^a_{b,s}$.

Whenever in this subsection we talk about sales we refer always to the point in time until 50% of the supplied items of the considered article are sold.

We define with

$$\hat{\delta}^a_b := \frac{sal50^a_b}{sal50^a} \cdot |B| \tag{3.1}$$

the *scaled relative demand* for branch b for Article a – scaled in such a way that the mean of $\hat{\delta}^a_b$ for Article a over all branches from the set B takes value one. Because not every observed product may be delivered to all branches this scaling is necessary to guarantee comparability of the observations.

With

$$\tilde{\delta}_b := \frac{\sum_{a \in \mathcal{A}} \hat{\delta}^a_b}{|\mathcal{A}|} \tag{3.2}$$

we define the *mean relative demand* for Branch b in terms of the set of articles $\mathcal{A}$.

The *mean amount of demand* δ_b for Branch b is given by the relative frequency

$$\delta_b := \frac{\tilde{\delta}_b}{\sum_{b' \in B} \tilde{\delta}_{b'}}. \tag{3.3}$$

Analogously we define the scaled relative demand for Size s in Branch b for Article a by

$$\hat{\delta}^a_{b,s} := \frac{sal50^a_{b,s}}{sal50^a_b} \cdot |S| \tag{3.4}$$

and the mean relative demand for Size s in Branch b for the set of articles $\mathcal{A}$ by

$$\tilde{\delta}_{b,s} := \frac{\sum_{a \in \mathcal{A}} \hat{\delta}^a_{b,s}}{|\mathcal{A}|}. \tag{3.5}$$

The mean amount of demand $\delta_{b,s}$ for Branch b and Size s is given by

$$\delta_{b,s} := \frac{\tilde{\delta}_{b,s}}{\sum_{s' \in S} \tilde{\delta}_{b,s'}}. \tag{3.6}$$

Finally, the *mean demand* $d_{b,s}$ for Branch b and Size s by

$$d_{b,s} = D \cdot \delta_b \cdot \delta_{b,s}. \tag{3.7}$$

That means we split up the estimated overall supply D to the particular sizes and branches in terms of the observed relative frequencies of sales.

Example 1 (relative demand estimation)**.** We want to illustrate the approach on a small example. We assume observations of sales according to the third column of the following table. We consider the case of three different observed articles and branches $B = \{b_1, b_2, b_3\}$. The scaled relative demands per Article a and Branch b are stated in the last column.

a	b	$sal50^a_b$	$\hat{\delta}^a_b$
a_1	b_1	6	1.50
a_1	b_2	2	0.50
a_1	b_3	4	1.00
a_2	b_1	3	1.00
a_2	b_2	4	1.33
a_2	b_3	2	0.66
a_3	b_1	2	0.86
a_3	b_2	2	0.86
a_3	b_3	3	1.29

In the next table in the second column we stated the mean relative demands per branch in terms of the set $\mathcal{A}$ of observed articles. In the third columns the corresponding mean amounts of demand are stated.

b	$\tilde{\delta}_b$	δ_b
b_1	1.12	0.37
b_2	0.90	0.30
b_3	0.97	0.33

In the following table in the second column the sales per article for the particular branches and sizes s_1, s_2, s_3, s_4 are stated. The corresponding scaled relative demands are stated in the fourth column.

a	b	$(sal50^a_{b,s_1}, sal50^a_{b,s_2}, sal50^a_{b,s_3}, sal50^a_{b,s_4})$	$(\hat{\delta}^a_{b,s_1}, \hat{\delta}^a_{b,s_2}, \hat{\delta}^a_{b,s_3}, \hat{\delta}^a_{b,s_4})$
a_1	b_1	(2,2,2,0)	(1.33,1.33,1.33,0.00)
a_1	b_2	(1,1,0,0)	(2.00,2.00,0.00,0.00)
a_1	b_3	(0,2,2,0)	(0.00,2.00,2.00,0.00)
a_2	b_1	(1,2,0,0)	(1.33,2.67,0.00,0.00)
a_2	b_2	(1,1,1,1)	(1.00,1.00,1.00,1.00)
a_2	b_3	(1,0,1,0)	(2.00,0.00,2.00,0.00)
a_3	b_1	(1,1,0,0)	(2.00,2.00,0.00,0.00)
a_3	b_2	(0,1,1,0)	(0.00,2.00,2.00,0.00)
a_3	b_3	(0,2,1,0)	(0.00,2.67,1.33,0.00)

The mean relative demands per branch and size are stated in the next table in the second column. In the third column the mean amount of demand per branch and size is stated.

b	$(\tilde{\delta}_{b,s_1}, \tilde{\delta}_{b,s_2}, \tilde{\delta}_{b,s_3}, \tilde{\delta}_{b,s_4})$	$(\delta_{b,s_1}, \delta_{b,s_2}, \delta_{b,s_3}, \delta_{b,s_4})$
b_1	(1.55,2.00,0.44,0.00)	(0.39,0.50,0.11,0.00)
b_2	(1.00,1.67,1.00,0.33)	(0.25,0.42,0.25,0.08)
b_3	(0.67,1.56,1.78,0.00)	(0.17,0.39,0.44,0.00)

On the basis of these computations for a given estimated overall supply of $D = 20$ we compute the mean demand $d_{b,s}$ per branch and size: For example, the mean demand for Size s_1 in Branch b_1 is given by $d_{b,s} = 20 \cdot 0.37 \cdot 0.39 = 2.89$. The mean demands for all branches and sizes are stated in the following table.

b	$(d_{b,s_1}, d_{b,s_2}, d_{b,s_3}, d_{b,s_4})$
b_1	(2.89,3.70,0.81,0.00)
b_2	(1.50,2.52,1.50,0.48)
b_3	(1.12,2.57,2.90,0.00)

3.2.2 Regarding different scenarios

As a next step we include the consideration of different scenarios. The resulting mean demand $d^e_{b,s}$ per branch b, size s and scenario e is applied in the SLDP, see Section 2.3.

Additionally we estimate the scenario probabilities $\mathrm{Prob}(e)$ which are applied in the SLDP and DISPO.

At first we determine the realized scenario for each observed article $a \in \mathcal{A}$. We observe sales until two weeks after sales start. For article $a \in \mathcal{A}$ the number of these sales over all branches and sizes is given by $sal2^a$ and the supply by $sup2^a$. The relation $rel2^a = \frac{sal2^a}{sup2^a} \in [0, 1]$ then indicates the popularity of the article. It is $rel2^a = 0$ if no item is sold in the first two weeks and $rel2^a = 1$ if all items are already sold out after two weeks. We categorize three different scenarios as they are stated in the following table.

$rel2^a$	e
< 0.33	low seller
$\geq 0.33, \leq 0.66$	normal seller
> 0.66	high seller

The scenario probabilities $\mathrm{Prob}(e)$ are given by the relative frequencies of the observed scenario in the historical data.

Now we determine how the demand for the low and the high scenario behaves against the normal scenario. This is done the following way:

We categorize the set of articles $\mathcal{A}$ by their scenarios.

For every scenario $e \in$ {low seller, normal seller, high seller} thus we obtain a set $\mathcal{A}_e$. The sales until 3 months after sales start for each article over all supplied branches and sizes are given by sal^a, the supply by sup^a. The *mean relative sales* rel^e over all articles for one particular scenario e are given by

$$rel^e := \frac{\sum_{a \in \mathcal{A}_e} \frac{sal^a}{sup^a}}{|\mathcal{A}_e|}. \tag{3.8}$$

We compute the *change of demand* df^e for scenario e as

$$df^e := \frac{rel^e}{rel^{\text{normal seller}}} \tag{3.9}$$

Thus, $df = 1$ for the normal seller scenario.

The mean demand for Scenario e, Branch b and Size s is given as

$$d^e_{b,s} := df^e \cdot d_{b,s}. \tag{3.10}$$

3.2.3 Splitting up the demand to sales periods

The next step is to split up the demand to the sales periods. This is done by estimating sales rates from the historical data for all periods – in our case weeks. With sal^a_k we denote the number of sales for Article a over all branches and sizes in Period k. We denote with sto^a_k the overall stock for Article a at the beginning of Period k. For Period k the *relative sales per period* rs^a_k for Article a are given by

$$rs^a_k := \frac{sal^a_k}{sto^a_k}. \tag{3.11}$$

The *mean relative sales per period* rs^a_k for Period k are given by

$$rs_k := \frac{\sum_{a\in\mathcal{A}} rs^a_k}{|\mathcal{A}|}. \tag{3.12}$$

The value rs_k equals the mean relative amount of sold items depending on the *stock at the beginning of Period* k. We now convert the $rs_k, k = 1, \ldots, k_{\max} - 1$ to a factor which describes the amount of sold pieces per size and branch depending on the *supply*.

We compute this factor, we call it the *sales rate per period* sr_k for Period k, by Algorithm 1.

Algorithm 1 Sales rates per period

Require: mean relative sales $rs_k, k \in K \setminus \{k_{\max}\}$
Ensure: sales rate $sr_k, k \in K \setminus \{k_{\max}\}$
1: init $sum = 0$
2: init $stock = 1$
3: **for all** $k = 0, \ldots, k_{\max} - 1$ **do**
4: $\quad nr = rs_k \cdot stock$
5: $\quad \tilde{sr}_k = nr$
6: $\quad stock = stock - nr$
7: **end for**
8: **for all** $k \in K \setminus \{k_{\max}\}$ **do**
9: $\quad sr_k = \frac{\tilde{sr}_k}{\sum_{j\in K} \tilde{sr}_k}$
10: **end for**

In the for-loop in Step 3 of Algorithm 1 the percentage remaining stock (beginning with a stock of one item or 100%) is computed according to the mean relative sales. From the current stock and the mean relative sales the values $\tilde{sr}_k$ arises. In Step 9 of the algorithm the values sr_k are computed by scaling the values $\tilde{sr}_k$ in such a way that the sum over all resulting sr_k takes value one.

Example 2 (sales rates)**.** In this example we assume $k_{\max} = 4$ that means 4 real sales periods. We assume that the historical data yields mean relative sales as they are stated below.

k	0	1	2	3
rs_k	0.5	0.7	0.2	0.4

Now we apply Algorithm 1. At first $stock$ is set to value one. In Period 0 according to rs_0 we sell 50% of the current stock or 0.5 items. The updated stock is set to 0.5 and $\tilde{sr}_0 = 0.5$. In Period 1 we start with a stock of 0.5. With $rs_1 = 0.7$, we assume that 70% of the current stock that means 0.35 items are sold. This yields a new stock of 0.15. It is $\tilde{sr}_1 = 0.35$. We proceed analogously for the last two periods and obtain $\tilde{sr}_2 = 0.03$ and $\tilde{sr}_3 = 0.048$.

We scale the values $\tilde{sr}_k$ according to Step 9 of Algorithm 1 and obtain the results stated in the following table.

k	0	1	2	3
sr_k	0.539	0.377	0.032	0.052

3.2.4 Price-dependent demand

An important factor in terms of our demand estimation is the influence of the sales price.

To estimate the impact of a mark-down in week k from price π_{p_1} to π_{p_2} with $p_2 > p_1$ the relative sales per week, as defined in the last subsection, for an observed article a in the week before the mark-down rs^a_{k-1} and in the week of the mark-down rs^a_k are compared. The observed increase of sales is given by the factor $\tilde{elas}^{\,a}_{\pi_{p_1}\to\pi_{p_2}} = \frac{rs^a_k}{rs^a_{k-1}}$. With $no^{\pi_{p_1}\to\pi_{p_2}}$ beeing the number of articles for which a mark-down from π_{p_1} to π_{p_2} was observed, the mean elasticity $elas_{\pi_{p_1}\to\pi_{p_2}}$ is given by $\frac{\sum_{a\in\mathcal{A}} \tilde{elas}^{\,a}_{\pi_{p_1}\to\pi_{p_2}}}{no^{\pi_{p_1}\to\pi_{p_2}}}$.

3.2.5 Combining the estimated factors

The scenario-, time- and price-dependent demand for a product with starting price π_0 for branch b and size s given by

$$d^e_{k,b,s,p} = d_{b,s} \cdot df^e \cdot sr_k \cdot \prod_{p'\in P: p'\le p} elas_{\pi_0\to\pi_{p'}}. \tag{3.13}$$

3.2.6 Updating the scenario

In the approach POP-RH, see Subsection 2.5.4, we use latest sales figures to determine the scenario in effect. Before we perform POP-RH to adapt our mark-down policy for the subsequent periods we update demand estimation by adapting the factor for the change of demand from Subsection 3.2.2. This is done by comparing predicted overall sales with realized overall sales. It is sal^{obs}_k the amount of sales over all branches and sizes for the last period k. The amount of predicted sales over all branches and sizes for Period k is given by sal^{real}_k. Then our updated change of demand df^{e_k} is given by $df^{e_k} = \frac{sal^{obs}_k}{sal^{real}_k}$.

We compute the dependent demands for the next period $k+1$ by

$$d^{e_k}_{k+1,b,s,p} = d_{b,s} \cdot df^{e_k} \cdot sr_{k+1} \cdot \prod_{p'\in P: p'\le p} elas_{\pi_{p_0}\to\pi_{p'}}. \tag{3.14}$$

3.3 Logistic regression

With the aim to compare the empirical estimation method outlined in the last chapter with a more common parametric approach we performed ordinal logistic regression. In Subsection 3.3.1 we outline the basic concepts of maximum likelihood estimation – this is the common approach to estimate logistic regression models. We introduce binary logistic regression in Subsection 3.3.2 before we extend it to ordinary logistic regression in Subsection 3.3.3. We are mainly guided by [Har10] and [Rya08].

3.3.1 Maximum likelihood estimation

Maximum likelihood estimation (MLE) is a common statistical method to estimate parameters in generalized linear models. The so-called *likelihood function* is defined as the joint probability function of the random variables.

Assumed we are given o observations $Y_i, i = 1, \dots, o$ and we want to estimate unknown parameters $\beta = (\beta_1, \dots, \beta_n)$. We denote with $f_i(y, \beta)$ the density function of the random variable y for the i-th observation. The likelihood for the i-th observation is given by $L_i(\beta) := f_i(Y_i, \beta)$.

With the assumption that the observations are independent from each other the *likelihood function* is given by

$$L(\beta) = \prod_{i=1}^{o} L_i(\beta). \tag{3.15}$$

The *log likelihood function* is the logarithm of the likelihood function:

$$\ln(L(\beta)) = \sum_{i=1}^{n} \ln(L_i(\beta)) \tag{3.16}$$

The *maximum likelihood* is the value of β that maximizes $\ln(L(\beta))$ as a function of β. It can be computed by considering the first derivates of $L_i(\beta)$, the so-called *score-vector*, i.e. the gradient, and the matrix of the second derivates, also known as *observed information matrix* i.e. the Hessian-matrix. For a maximizing β all first partial derivates have to take value zero while the observed information matrix has to be negative definite.

The log likelihood function is applied because in much cases the derivate of the log likelihood function – because of the properties of the logarithm, for example changing from product to sum – is easier to compute as the likelihood itself. And if β maximizes the log-likelihood function then it also maximizes the likelihood function.

If it is not possible to determine the maximum likelihood exactly, so-called iterative trial-and-error methods are used. The most common method is the so-called Newton-Raphson or simply Newton method. This method is a standard approach in nonlinear optimization. The score vector $U(\beta)$ is locally approximated by a linear function of β in a small region. In the following we denote with $I(\beta)$ the observed information matrix. With a starting estimate of $\beta^{(0)}$ of the maximum likelihood β the linear approximation is given by

$$U(\beta) = U(\beta^{(0)}) - I(\beta^{(0)})(\beta - \beta^{(0)}). \tag{3.17}$$

Equating to 0 and solving by β yields

$$\beta = \beta^{(0)} + I^{-1}(\beta^{(0)})U(\beta^{(0)}). \tag{3.18}$$

Thus, we determined the null of the linear approximation of the maximum likelihood function in β. At the i-th step we obtain the next estimate by

$$\beta^{(i+1)} = \beta^{(i)} + I^{-1}(\beta^{(i)})U(\beta^{(i)}). \tag{3.19}$$

If it is the case that the log likelihood worsened at $\beta^{(i+1)}$ then "step-halfing" is applied, that mean $\beta^{(i+1)}$ is replaced by $\frac{\beta^{(i)}+\beta(i+1)}{2}$. This is done until the likelihood still is worse than the likelihood at $\beta^{(i)}$. The method is iterated until convergence.

For more details about maximum likelihood estimation see for example [Har10] or [Rya08]. For further reading in terms of the Newton method, we refer to [Rus06].

3.3.2 Binary logistic Regression

The assumption for binary logistic regression is, that the outcome – the dependent y-variable only takes value zero or one. Independent variables $x_i, i = 1, \ldots, k$ may be binary, integer, continuous or multinomial.

We denote the probability that the outcome y takes value one under the observations $x_i, i = 1, \ldots, k$ by $P(y = 1|x_1, x_2, \ldots, x_k) =: P(y = 1)$. Analogously the probability for y taking value zero is denoted by $P(y = 0|x_1, x_2, \ldots, x_k) =: P(y = 0)$.

Binary logistic regression tries to estimate these probabilities.

For this purpose the so called *odds* is defined as

$$odds := \frac{P(y = 1)}{1 - P(y = 1)} = \frac{P(y = 1)}{P(y = 0)} \tag{3.20}$$

where both probabilities $P(y = 1)$ and $P(y = 0)$ should not take value zero.

The odds which takes values $\in [0, \infty[$ is transformed to the so-called *logit* by the logarithmic function.

$$logit := \ln(odds) = \ln\left(\frac{P(y = 1)}{1 - P(y = 1)}\right) \tag{3.21}$$

In contrast to the odds the logit can take all real values.

The logit is estimated by a linear function

$$\ln\left(\frac{P(y = 1)}{1 - P(y = 1)}\right) = \alpha + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_k x_k. \tag{3.22}$$

Rearranging Equation (3.22) yields

$$P(y = 1) = \frac{1}{1 + \exp(-(\alpha + \sum_{i=1}^{k} \beta_i x_i))}. \tag{3.23}$$

where the right side is known as the *logistic function*.

This is a nonlinear function. The coefficients α and $\beta_i, = 1, \ldots, n$ are estimated by maximum likelihood estimation.

Because there are only two possibilities for the outcomes – zero or one – each observation can be seen as a Bernoulli experiment. For a sample of size n and observations $o_1, \ldots, o_n$ the likelihood function therefore is given by

$$L(\beta) = \prod_{i=1}^{n} P_i^{o_i}(1 - P_i)^{1-o_i}. \tag{3.24}$$

where it is $P_i = P(o_i = 1) = \frac{1}{1+\exp(\sum_{i=1}^{k} \beta_i x_i)}$.

3.3.3 Ordinal logistic regression

If the dependent variable y can also take other values than zero or one the binary logistic regression model may be extended to an ordinal logistic regression model. The restriction is that the possible outcomes stand together in an ordinal relationship. If this is not the case one would favor a *multinomial* logistic regression model. The proceeding in that case is different. For further reading about multinomial regression see for example [Har10].

We assume levels $j = 0, 1, \ldots, j_{max}$ for the dependent variable. Now the odds for $j = 0, 1, \ldots, j_{max}$ is defined as

$$odds = \frac{P(y \geq j)}{1 - P(y \geq j)} \tag{3.25}$$

and the logit as

$$logit = \ln\left(\frac{P(y \geq j)}{1 - P(y \geq j)}\right) = \alpha_j + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_k x_k. \tag{3.26}$$

This yields the following coherence:

$$P(y \geq j) = \frac{1}{1 + \exp(-(\alpha_j + \sum_{i=1}^{k} \beta_i x_i))} \tag{3.27}$$

In this model the regression coefficients β_i are independent of j. Only the α_j which describe the intercept depend on j. For a specific j the formulation is consistent with the binary regression model, see (3.23).

The ordinal model can also be estimated via maximum likelihood estimation. For detailed information we refer the reader to [McC80].

3.4 Applying ordinal logistic regression

Now we apply ordinal logistic regression to estimate the sales per period$\hat{=}$week, size, branch and price at our industrial partner. For this purpose we use the environment GNU R for statistical computing[2].

3.4.1 Data sample

In our first trials GNU R in terms of memory could not deal with the transaction data of the hole tested commodity group. Thus, we will perform all tests on a smaller data set. We consider historical data from the commodity group "women overgarments classic" containing transaction data for about one year starting in September 2009. We restrict the set of about 1400 branches by randomly choosing 30 branches and the set of sizes by randomly choosing 3 sizes. (Mostly there are 6 or 7 different sizes per article in this commodity group.) Moreover we will only regard articles which can be observed at least 13 weeks beginning from the sales start. We divided the remaining 116 articles randomly aiming at ratio $70 : 30$. This yields a set of 81 articles for estimation and a set of 35 articles for validation.

We categorize our observations in observations per article, week, branch and size. For our test set of 81 articles this results in $81 \cdot 13 \cdot 30 \cdot 3 = 94\,770$ data points.

[2] GNU R version 2.10.2

Frequencies of Responses				
0	1	2	3	4
90326	4118	299	17	10

Table 3.1: Estimation – Responses

Analogously we get $40\,950$ observations for the validation set.

In our data set there are five types of responses. The dependent variable, the number of sales, takes value 0,1,2,3 or 4. The frequencies are stated in Table 3.4.1.

3.4.2 Choosing the model

Analogously to the empirical estimation we include popularity of the article, week, price, size and branch in the estimation and interpret the branches and sizes as categorical incomes. In contrast to the empirical method now we also consider current stock as input. At this point we want to estimate realized sales and not demand. We have to decide on how the price is adopted in the estimation. We are given the starting price and the realized price per week. We tried three different models in terms of the adoption of the price which include

1. only realized price (Model 1),
2. only ratio realized price divided by starting price (Model 2),
3. ratio and starting price (Model 3).

as independent variables.

We will present the complete Model 3 before we comment on statistical tests in terms of the model fitness for all models.

We denote by sa the number of sold items and by at the popularity of the corresponding article. Analogously to the empirical estimation the popularity is computed by dividing the sales of the first two weeks by the stock at the beginning of the first week. With we we denote the corresponding week and with st the stock at beginning of the week and by pr the sales price and by sp the starting price. The additional binary variables $\{c_b\}, \{c_s\}, \{c_{b,s}\}$ take value one if and only if the observation concerns Branch b and/or Size s.

The probability $P(sa \geq j)$ with $j > 0$ after Model 3 is given as follows ($P(sa \geq 0)$ by definition takes value one):

$$P(sa \geq j) = \frac{1}{1 + \exp(-\alpha_j - \beta X)}, \tag{3.28}$$

where

$$\beta X = \beta_{at}\, at + \beta_{we}\, we + \beta_{st}\, st + \beta_{sp}\, sp + \beta_{pr} \frac{pr}{sp} + \sum_{b \in B} \beta_b \{c_b\} + \sum_{s \in S} \beta_s \{c_s\} + \sum_{\beta \in B} \sum_{\beta \in S} \{c_{b,s}\}$$

We estimate the coefficients of the model under use of GNU R. For maximum likelihood estimation we used the function lrm (logistic regression model) with default settings from the Design-package which was implemented by Frank Harrell, see also [Har10]. To evaluate the fitness of the model we state results from some statistical tests. These tests also were performed under use of the Design-package.

The output of statistical tests in terms of the model fitness for our three models are stated in Table 3.2. We will state some general information about statistical tests in Subsection 10.1.2. For more detailed information we refer the reader to [FPP07].

One of the performed tests is the so-called *likelihood-ratio test*, see for example [Har10]. For this a so-called *null-model* is compared to the estimated model. Null-model means that all parameters are set to zero. Thus, we compare if our model yields better results than they could arise by pure chance. If L_0 is the likelihood of the null-model and L_1 the likelihood of our estimated model, then the likelihood-ratio test statistic is given by $-2\log(L_0/L_1)$. The related test statistics for our models are stated in the third column of Table 3.2. This test statistic follows a χ^2-distribution as test distribution, see for example [FPP07] for further informations about this topic. In the second column with name "Max Deriv" the maximum absolute value of the first derivate of the logarithmized likelihood function , also known at Fisher score, is stated. The first derivate numerically takes value zero at a maximum. The degrees of freedom for the χ^2-distribution are stated in the fourth column "d.f". This value equals the number of estimated parameters. The *null-hypothesis* H_0 of the likelihood-ratio test says the estimation does not yield a better result than the null model, i.e. it is random. The *alternative hypothesis* H_1 says it does. The related *p-value* is stated in the fifth column with name "P". It equals the probability to obtain the observed test statistic or a lower one by chance. If the probability lies under a predefined significance level – in general 5% – then we deny the null-hypothesis, the result is *significant*: The probabilities to obtain an equal or better result for our models than by chance in all cases are small enough to trust the alternative hypothesis.

A further measurement for the fitness of the model is the so-called Nagelkerle Pseudo-R^2 index. It is defined by

$$R^2 := \frac{1 - \frac{L_0}{L_1}^{\frac{2}{n}}}{1 - L_0^{\frac{2}{n}}} \tag{3.29}$$

where n is the sample size. There are also other R^2 indices like for example McFaddens Pseudo-R^2 or Cox-Snell Pseudo-R^2. But these values dependent on the sample size and it is difficult to give a general statement about which value is good or bad. The advantage of Nagelkerle's version is that the value lies – independent from the number of observations – in between zero and one. We can assess a perfect "model fit" if R_2 takes value one. Thus, an index of value 0.15 for Model 3 is not as good.

The seventh column "Dxy" of Table 3.2 concerns the result of the so-called *Somers' D* test [Som62]. The test measures the coherence of the observations and the estimated values for the dependent variable. For this test we consider so-called *concordant* and *discordant* pairs and *ties*. With two distinguishing observations o_1 and o_2 and the two estimated responses r_1 and r_2 with probabilities $p(r_1)$ and $p(r_2)$ we call the pair concordant if $o_1 < o_2$ and $p(r_1) < p(r_2)$ or $o_1 > o_2$ and $p(r_1) > p(r_2)$. Otherwise the pair is called discordant. If the predicted probabilities are numerically equal the pair is called tied. If nc is the number of concordant pairs, nd the number of discordant pairs and nt the number of ties, then the Somers' D rank order correlation statistics D_{xy} is given by

$$D_{xy} := \frac{nc - nd}{nc + nd + nt}. \tag{3.30}$$

This value can lie in between -1 and $+1$. A negative value would evidence that the direction of prediction is inaccurate for the most pairs, a positive value that it is accurate. Our predicted value of 0.573 for Model 3 speaks for accurateness for the most pairs.

Model	Obs	Max Deriv	Model L.R.	d.f.	P	C	Dxy	R2	Brier
1	94770	2e-11	4870.02	93	0	0.786	0.573	0.15	0.043
2	94770	6e-12	4667.63	93	0	0.783	0.566	0.144	0.043
3	94770	1e-11	4870.3	94	0	0.787	0.573	0.15	0.043

Table 3.2: Evaluation of Model 1

An similar measurement is Harrell's concordance index C [Har10]. It is given by

$$C := \frac{D_{xy} + 1}{2} \tag{3.31}$$

in the sixth column of the table with value 0.787 for Model 3 and scales the Somers' D correlation to a value between 0 and 1. A value of 0.5 means that the direction of prediction neither is accurate nor inaccurate, a value of 1 high accurateness.

At the last column of the table the so called Brier score Br is stated. Brier score is a similar measurement as the mean-squared standard error, see for example [FPP07] and is defined as

$$Br := \frac{1}{n} \sum_{i=1}^{n} (p(r_i) - o_i)^2 \tag{3.32}$$

for n observations o_i and responses r_i with probabilities $p(r_i)$. Brier score measures the deviation between predicted and observed values. If the forecast was perfect, then the Brier score would take value zero. The smaller the Brier score is the better. For Model 3 the Brier score takes value 0.043.

We see that for Model 3 we obtain equal or a little better results in terms of the performed statistical tests than for the other models. Thus, we decided to use Model 3 for estimation.

3.4.3 Result

In the following $\{40\}$,$\{44\}$ denote the considered sizes, the remaining $\{\dots\}$ the branches. The result is adjusted to Size 38 and to a branch with name 255. Therefore these variables do not appear in the following result. Estimation of Model 3 yields

$$\text{Prob}\{sa \geq j\} = \frac{1}{1 + \exp(-\alpha_j - X\beta)}, \text{ where}$$

$$\begin{aligned} \hat{\alpha}_1 &= -2.991214, \\ \hat{\alpha}_2 &= -5.724494, \\ \hat{\alpha}_3 &= -8.236504, \\ \hat{\alpha}_4 &= -9.233339, \end{aligned}$$

$$
\begin{aligned}
X\hat{\beta} = \\
&+1.266487\ at - 0.05551025\ we + 0.8386404\ st \\
&-0.1073931\ sp - 1.025878\ \tfrac{pr}{sp} \\
&+0.40555481\{683\} + 0.42276266\{701\} - 0.06807982\{894\} + 0.91453736\{1096\} \\
&+1.14605624\{1160\} + 0.51938412\{1224\} + 0.04119262\{1375\} + 0.62334482\{1384\} \\
&+0.11655906\{1395\} + 0.10298705\{1432\} + 0.42631106\{1456\} + 0.78788775\{1484\} \\
&+0.41849692\{1486\} + 0.40607731\{1490\} + 0.73817205\{1527\} + 0.85198384\{1569\} \\
&+0.55776877\{1599\} + 0.34085792\{1687\} + 0.06353926\{1720\} + 0.68430028\{1863\} \\
&+0.66197427\{1882\} + 0.30272593\{1929\} + 1.00467592\{1964\} + 0.74825459\{1991\} \\
&+0.17367353\{2056\} + 0.65254159\{2066\} + 0.42737426\{2093\} + 0.68953835\{2096\} \\
&+0.24661423\{2182\} \\
&+0.5022497\{40\} + 0.5660673\{44\} \\
&+\{40\}[-0.1679237\ \{683\} - 0.2582429\ \{701\} - 0.4211826\ \{894\} - 0.6175968\ \{1096\} \\
&-0.6293785\ \{1160\} - 0.8027621\ \{1224\} - 0.6984557\ \{1375\} - 0.5685319\ \{1384\} \\
&-0.2725717\ \{1395\} - 0.1415956\ \{1432\} - 0.3708545\ \{1456\} - 0.5452929\ \{1484\} \\
&-0.8955414\ \{1486\} - 0.4984089\ \{1490\} - 0.2372495\ \{1527\} - 0.6378074\ \{1569\} \\
&-0.4431316\ \{1599\} - 0.8218475\ \{1687\} - 0.6595380\ \{1720\} - 0.6408166\ \{1863\} \\
&-0.5815683\ \{1882\} - 0.3830782\ \{1929\} - 0.8092480\ \{1964\} - 0.7050292\ \{1991\} \\
&-0.2009142\ \{2056\} - 0.7501039\ \{2066\} - 0.2857802\ \{2093\} - 0.4810097\ \{2096\} \\
&-0.7264964\ \{2182\}] \\
&+\{44\}[-0.3505462\ \{683\} - 0.7450688\ \{701\} - 0.1731708\ \{894\} - 0.3135945\ \{1096\} \\
&-0.7559348\ \{1160\} - 0.8709310\ \{1224\} - 0.5686853\ \{1375\} - 0.3835397\ \{1384\} \\
&-0.8152310\ \{1395\} - 0.1074382\ \{1432\} - 0.1655254\ \{1456\} - 0.6748133\ \{1484\} \\
&-0.6939229\ \{1486\} - 0.2487922\ \{1490\} - 0.4253804\ \{1527\} - 0.5045935\ \{1569\} \\
&-0.1982682\ \{1599\} - 0.8017821\ \{1687\} - 0.1931474\ \{1720\} - 0.7426981\ \{1863\} \\
&-0.5992249\ \{1882\} - 0.2742510\ \{1929\} - 0.6623949\ \{1964\} - 0.4209326\ \{1991\} \\
&-0.2403357\ \{2056\} - 0.6230690\ \{2066\} - 0.8306514\ \{2093\} - 0.4560512\ \{2096\} \\
&-0.7413656\ \{2182\}] \\
&\text{and}\{c\} = 1 \text{ if subject is in group } c,\ 0 \text{ otherwise.}
\end{aligned}
$$

The estimated parameters are partly stated in Table 3.3, more precisely in the second column. The remaining parameters can be found in Appendix D. We see that popularity and current stock have a positive effect on the sales probabilities while the sales week and the price have a negative effect. This is also reproduced by the in Section 3.2 described empirical estimation method.

With the so-called *Wald test* [Wal43] it is tested if particular parameters have a significant influence on the estimation, For the computation of the Wald statistic we consider a model where the considering variable is missing – similar to the null model, but all other independent variables are contained. The test-statistic is defined as the deviation – in units of the standard error – from the mean for the estimated model to the mean of the model where the corresponding coefficient takes value zero. It follows a standard normal distribution as test distribution. The null-hypothesis is that our model does not yield a better estimation than the model where this coefficient takes value zero, the alternative says it does.

If the Wald statistic would yield no significant result than the variable could be omitted from the model because the influence is not significant. The standard error is stated as "S.E." in the third column of the table, the test statistic – the so-called *z-score* "Wald Z" in the fourth column. In the fifth column the corresponding p-value as "P" is

	Coef	S.E.	Wald Z	P
$y \geq 1$	-2.99121	0.256641	-11.66	0.0000
$y \geq 2$	-5.72449	0.262209	-21.83	0.0000
$y \geq 3$	-8.23650	0.320577	-25.69	0.0000
$y \geq 4$	-9.23334	0.407111	-22.68	0.0000
at	1.26649	0.063337	20.00	0.0000
we	-0.05551	0.004702	-11.81	0.0000
st	0.83864	0.016701	50.22	0.0000
sp	-0.10739	0.007625	-14.08	0.0000
$\frac{pr}{sp}$	-1.02588	0.092116	-11.14	0.0000
branch $= 683$	0.40555	0.251831	1.61	0.1073
branch $= 701$	0.42276	0.294544	1.44	0.1512
branch $= 894$	-0.06808	0.327038	-0.21	0.8351
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$

Table 3.3: Estimated parameters

stated. We see that popularity, week, stock and price have a significant influence on the estimation. For some branches and sizes this is not the case: The behavior is similar to that of the reference branch “38” or the reference size “255’. See Appendix D for all estimated parameters.

3.5 Comparison of different estimation methods

As already mentioned we applied the empirical estimation method in DISPO. All results in this thesis rely on this method. This raises the question of whether there would have been a better performance by applying a standard parametric approach like logistic regression. In the last section we mentioned some statistical tests to assess the model fitness for the logistic regression model. But these tests cannot be applied on the empirical method. So we had to implement a different strategy: We compare the methods on historical transaction data. But how can we deal with lost sales?

3.5.1 Methodology

A “good” method should predict demand or sales as accurately as possible. In the end this should lead on to a supply policy for which the realized revenue is possibly high. So we will assess our methods in terms of the revenue that would arise by a supply according to the estimates.

For both estimation methods we determine revenue-maximizing independent supplies per branch and size according to the predicted demand/sales[3]. Our predictions in terms of the logistic regression model are the related (branch,size,price,stock) expectational values $\sum_{j=0}^{4} [P(y = j) \cdot j]$.

Because in the empirical demand estimation the overall number D of supplied items is needed as input in both cases we only allow solutions for which the supply over all observed branches and sizes equals the observed overall supply. Otherwise the empirical estimation would have an advantage over the other method. That means we redistribute the supplied items according to our estimations. For the comparison we take a pass on the lot-type restriction because it would distort our results – at this

[3]Therefor we use the later outlined Algorithm 8 in correspondence with Algorithm 6 to compute the so-called single supply revenues for the in the historical data observed prices, see Section 7.1.

point we are interested in the goodness of the predictions and not in how they could be implemented by a supply in terms of lot-types.

Because of the right-censored transaction data, a difficulty we have to deal with is the fact that we can only observe sells in the data if the current stock in the considered branch and size does not take value zero – lost sales may occur. If the stock (observed in the historical data) in Period k amounts to zero we have to regard that possibly more items would had been sold in this or the subsequent periods if it would have been possible. It describes $\tilde{r}_f^{\text{model}}$ the stock at the beginning of period f (for the considered branch and size) if a supply according to the estimation of the related model had been applied. With d_f we denote the related demand for Period f. It is $\tilde{a}_f^{\text{model}}$ the discounted revenue that would have been arisen and $\tilde{t}_f$ the price in period f. For all periods f with $0 < f < k$ we can compute the related revenue we would have earned by supplying according to the model by

$$\tilde{a}_f^{\text{model}} = \tilde{a}_{f-1}^{\text{model}} + \tilde{t}_f \cdot \min\{\tilde{r}_f^{\text{model}}, d_f\}.$$

The updated stock after Period f is the stock at the beginning of Period f minus the sales at Period f:

$$\tilde{r}^{f+1} = \max\{0, \tilde{r}^f - d_f\}.$$

For Period k the proceeding is different. There are three possibilities:

1. the current stock does not differ for the both methods,
2. the current stock for the first estimation method is higher than the current stock for the second one,
3. the current stock for the second estimation method is higher than the current stock for the first one.

In the first case we can neglect potential lost sales. For both methods they would yield the same revenue. In the second case we compute the value $ri =$"stock at k in terms of the first method minus stock at k in terms of the second method". The difference equals the number of items which might have been additionally sold if Method 1 compared with Method 2 had been used for estimation. The maximum *additional revenue* for Method 1 would have been arisen if the ri items had been still sold in Period k. Due to the non-increasing prices it is the discounted revenue for selling all the remaining items ri in Period k. In contrast the revenue that is implied by observed sales is called *sure revenue*.

Analogously we precede in the third case. Here the second estimation method may lead to additional revenue.

We call Estimation method 1 for an article *surely better* than Method 2, if the sure revenue for Method 1 is higher than the sure revenue plus the additional revenue for Method 2.

3.5.2 Results

In Table 3.4 we listed the results of comparing the empirical method with the logistic regression model according to the described methodology. We performed the test for 34 articles. In the first column the overall supplies are listed. The sure revenues for the empirical method "e" and the ordinal logistic regression model "l" are stated in the next two columns, the maximum additional revenue in columns four and five. The entries

	supply	sure rev.		max add. rev		surely better		equal
		e	l	e	l	e	l	
	141	60.85	60.85	0.00	0.00	0	0	1
	109	177.76	176.83	7.04	0.00	1	0	0
	145	186.43	174.49	0.00	0.00	1	0	0
	153	−56.24	−56.24	0.00	0.00	0	0	1
	93	−53.56	−53.56	0.00	0.00	0	0	1
	85	77.38	82.33	13.49	13.49	0	0	0
	82	64.63	64.63	26.64	26.64	0	0	1
	142	284.23	232.01	18.48	98.88	0	0	0
	109	−44.39	−44.39	0.00	0.00	0	0	1
	157	19.51	19.51	0.00	0.00	0	0	1
	51	29.61	18.50	50.23	57.40	0	0	0
	93	126.44	126.44	0.00	0.00	0	0	1
	128	10.37	10.03	3.64	0.00	1	0	0
	129	−73.65	−73.65	0.00	0.00	0	0	1
	113	499.77	413.67	111.89	243.41	0	0	0
	122	−23.92	−23.92	0.00	0.00	0	0	1
	79	116.61	101.30	21.53	28.70	0	0	0
	120	3.11	3.11	0.00	0.00	0	0	1
	121	−225.79	−225.79	0.00	0.00	0	0	1
	137	−20.23	−59.06	122.08	411.54	0	0	0
	136	−278.14	−278.14	0.00	0.00	0	0	1
	68	144.82	172.18	21.53	7.18	0	1	0
	89	245.93	237.06	17.76	26.64	0	0	0
	82	76.69	80.27	14.58	10.93	0	0	0
	54	30.18	26.24	18.22	18.22	0	0	0
	130	57.40	22.43	9.24	0.00	1	0	0
	61	87.75	77.58	35.52	35.52	0	0	0
	14	−21.19	14.68	57.41	21.53	0	0	0
	64	101.66	107.16	25.30	18.97	0	0	0
	109	127.44	87.85	25.47	68.52	0	0	0
	134	61.13	27.66	3.61	69.25	0	0	0
	102	−39.73	−39.73	0.00	0.00	0	0	1
	70	80.94	88.09	28.70	21.53	0	0	0
	80	6.34	−0.77	31.62	31.62	0	0	0
$\sum$	3500	1840.14	1569.65	663.98	1209.97	4	1	13
$\varnothing$	103	54.12	46.17	19.53	35.59	11.76%	2.94%	38.24%

Table 3.4: Comparison of empirical estimation with logistic regression

in the three last columns indicate if the considered estimation method is surely better than the other one or if both methods perform equal in terms of the sum sure revenue plus additional revenue.

In 11.76 percent that means 4 of the considered cases we see that the empirical estimation is surely better than the logistic regression model. In contrast we get only one win, this equals 2.94 percent wins, for the logistic regression method. In 13 of the 34 cases or 38.24 percents both methods would yield the same revenue. Although we are not able to attest better predictions to our empirical approach for sure there is obviously no reason for us to replace it by the logistic regression model.

Chapter 4

Price Optimization

In this chapter we elaborate on the Price Optimization Problem, how it takes part in DISPO and present algorithms to solve it. We restrict us to the case that the set of scenarios contains only one scenario. This is the case when we perform price optimization with receding horizon POP-RH, Subsection 2.5.4: The set of scenarios consists only of the current scenario in effect.

In Section 4.1 we introduce a mixed-integer programming formulation for the Price Optimization Problem for one fixed scenario $\hat{e}$, the POP$^{\hat{e}}$. To describe the situation at our industrial partner as correctly as possible, we extend price optimization as it is performed by the former DISPO-team by adding mark-down costs depending on the current stock. This leads to nonlinearity of the underlying mixed-integer program.

We adapt the mark-down policy every period/week, see Subsection 2.5.4 or Figure 1.1. The last sales for the week can be observed on Saturday evening. On Monday morning already the decision for mark-downs has to be made. At our industrial partner weekly more than 4000 articles have to be considered. Therefore the solving process of price optimization must not last too long.

Even without regarding stock-depending costs for mark-downs solving price optimization with state-of-the-art MIP solvers – because of the long computation times – is not suitable in terms of these real-world requirements.

In the MIP formulation of the POP$^{\hat{e}}$ the variables are finely grained: For every period and price index there exists a binary variable that indicates if the related price is assigned to the period or not. One idea would be to enumerate all possible combinations of these variables – the so-called price trajectories, Section 4.2.

In Section 4.3 we introduce some basics of dynamic programming which we apply in Section 4.4 on the POP-RH: We outline how to generate price trajectories for a given supply dynamically. For this purpose we develop dominance rules to exclude truncated price trajectories which will not lead to an optimal solution from further consideration, Section 4.5. The result is a so-called label setting algorithm for POP$^{\hat{e}}$. The detailed implementation is outlined in Section 4.6. We illustrate the algorithm on a small example, Section 4.8, that we introduce in Section 4.7 and that will accompany us in the remainder of the thesis. We state computational results in Section 4.9 and conclude the chapter in Section 4.10.

We will apply the enumeration of price trajectories in our exact Branch&Bound solver for ISPO which is presented in Section 9.1. The dynamic generation of price trajectories – besides price optimization with receding horizon, POP-RH – takes place in our heuristic approach for ISPO, see Section 9.2.

4.1 Extending POP by mark-down costs – a mixed-integer nonlinear program

In this section we state price optimization for one fixed scenario $\hat{e}$, Subsection 4.1.1. We will not go into details in terms of the problem specification at this point and refer the reader to Problem 5. Because we fix scenario $\hat{e}$ in the POP$^{\hat{e}}$ we drop the scenario index for the corresponding variables at this point. In Subsection 4.1.2 we go into the case that mark-down costs depending on the corresponding stock arise as it is the case at our industrial partner. Inclusion of these costs leads to nonlinearity of the introduced model.

4.1.1 Problem formulation

We formulate the Price Optimization Problem for a fixed scenario $\hat{e}$ with the inclusion of mark-down costs as follows:

Problem 5 (POP$^{\hat{e}}$)**.**

$$\max \sum_{k\in K} \exp(-\rho k)\Big(\sum_{b\in B}\sum_{s\in S} r_{k,b,s} - \mu_k \beta_k\Big) \tag{4.1}$$

subject to

$$\sum_{p\in P} u_{k,p} = 1 \quad \forall k \in K, \tag{4.2}$$

$$u_{k,0} = 1 \quad \forall k \in K : k < k_{\text{obs}}, \tag{4.3}$$

$$u_{k_{\max},p_{\max}} = 1, \tag{4.4}$$

$$u_{k-1,p_1} + u_{k,p_2} \leq 1 \quad \forall k \in K : k > 0, p_1, p_2 \in P : p_2 < p_1, \tag{4.5}$$

$$\beta_k \geq u_{k-1,p_1} + u_{k,p_2} - 1 \quad \forall k \in K : k > 0, \forall p_1, p_2 \in P : p_2 \neq p_1, \tag{4.6}$$

$$v_{0,b,s} = I_{b,s} \quad \forall b \in B, s \in S, \tag{4.7}$$

$$v_{k-1,b,s} - v_{k,b,s} = \sum_{p\in P} w_{k-1,b,s,p} \quad \forall k \in K : k > 0, b \in B, s \in S, \tag{4.8}$$

$$\sum_{p\in P} w_{k,b,s,p} \leq v_{k,b,s} \quad \forall k \in K, b \in B, s \in S, \tag{4.9}$$

$$w_{k,b,s,p} \leq u_{k,p} \cdot d_{k,p,b,s} \quad \forall k \in K : k < k_{\max}, b \in B, s \in S, p \in P : p < p_{\max}, \tag{4.10}$$

$$w_{k_{\max},b,s,p_{\max}} = v_{k_{\max},b,s} \quad \forall b \in B, s \in S, \tag{4.11}$$

$$r_{k,b,s} = \sum_{p\in P} \pi_p \cdot w_{k,b,s,p} \quad \forall k \in K, b \in B, s \in S, \tag{4.12}$$

$$u_{k,p} \in \{0,1\} \quad \forall k \in K, p \in P, \tag{4.13}$$

$$\beta_k \in \{0,1\} \quad \forall k \in K, p \in P, \tag{4.14}$$

$$w_{k,b,s,p} \geq 0 \quad \forall k \in K, b \in B, s \in S, p \in P, \tag{4.15}$$

$$v_{k,b,s} \geq 0 \quad \forall k \in K, b \in B, s \in S, \tag{4.16}$$

$$r_{k,b,s} \geq 0 \quad \forall k \in K, b \in B, s \in S. \tag{4.17}$$

A mark-down in period k is indicated by the dependent binary variable β_k, which is forced to one by Inequality (6.19) if the price compared to the previous period has changed. In the objective the mark-down costs μ_k for every period are subtracted from the revenue.

4.1.2 Nonlinearity by mark-down costs

We want to put a finer point to the mark-down costs μ_k for Period k.

Mark-down costs divide into two parts. On the one side there are fixed mark-down cost. If there is a mark-down in a period always cost of μ_f occur independent from the number of items that have to be marked down. On the other side there are variable mark-down cost: Every single item that has to be marked down causes cost of μ_v.

At our partner fixed mark-down costs arise from all actions that are necessary to inform the branches about mark-downs, the variable mark-down costs accrue from pricing the items in the branches by the sales personnel.

There is an exception for the last/sellout period $k_{\max}$. We assume that in the sellout process still $q_{k_{\max}}$ mark-downs are necessary. Only variable mark-down costs arise for the sellout period.

Altogether the mark-down costs μ_k in the real sales period k are given by

$$\mu_k = \mu_f + \mu_v \sum_{b \in B} \sum_{s \in S} v_{k,b,s}, \forall k \in K \setminus \{k_{\max}\}. \tag{4.18}$$

For the sellout period $k_{\max}$ the mark-down costs are given by

$$\mu_{k_{\max}} = q_{k_{\max}} \mu_v \sum_{b \in B} \sum_{s \in S} v_{k_{\max},b,s}. \tag{4.19}$$

Extending the formulation of Problem 5 by Constraints (4.18) and (4.19) leads to a mixed-integer nonlinear program because in the objective function we have to multiply the binary variables β_k via μ_k with the real variables $v_{k,b,s}$.

In the remainder we will regard the mark-down costs as formulated by the constraints (4.18) and (4.19). Henceforth, whenever we will mention the Price Optimization Problem, we refer to the POP$^{\hat{e}}$

4.2 Enumerating price trajectories

We can enumerate the possible price trajectories by branching on the decisions of the price optimization stage. The variables for the price optimization stage are finely grained – for every period k and price index p there is a binary variable $u_{k,p}$ which indicates whether the price with index p is assigned to k. Now we consider more widescale decisions. A natural idea is to condense the mark-down decisions in each time period to an entire price trajectory for the complete selling time.

Definition 1 (price trajectory, revenue of a price trajectory)**.** We define a *price trajectory* $t = (t_0, \ldots, t_{k_{\max}})$ as a $k_{\max} + 1$-tuple where each entry is a price index $p \in P$. It is $t_k = 0$ for $k < k_{\text{obs}}$ and $t_{k_{\max}} = p_{\max}$. Moreover it is $t_k \leq t_{k+1}, \forall k \in K \setminus \{k_{\max}\}$. That is a price trajectory equals a valid assignment of the $u_{k,p}$ variables in Problem 5 where $t_k = p$ if and only if $u_{k,p} = 1$. The *revenue* a resulting from a price trajectory

which equals the related objective value of Problem 5 is given by

$$
\begin{aligned}
a := \sum_{k=0}^{k_{\max}-1} \exp(-\rho k) \Bigg(& \pi_{t_k} \sum_{b\in B} \sum_{s\in S} \min \left\{ \max \left\{ I_{b,s} - \sum_{j=0}^{k-1} d_{j,b,s,t_j}, 0 \right\}, d_{k,b,s,t_k} \right\} \\
& - \beta_k \left(\mu_f + \mu_v \sum_{b\in B} \sum_{s\in S} \max\{I_{b,s} - \sum_{j=0}^{k-1} d_{j,b,s,t_j}, 0\} \right) \Bigg) \\
& + \exp(-\rho k_{\max})(\pi_{k_{\max}} - q_{k_{\max}} \mu_v) \cdot \sum_{b\in B} \sum_{s\in S} \max \left\{ I_{b,s} - \sum_{k=0}^{k_{\max}-1} d_{k,b,s,t_k}, 0 \right\}
\end{aligned}
\tag{4.20}
$$

According to the constraints (4.7), (4.8), (4.9) and (4.10) $\max\{I_{b,s} - \sum_{j=0}^{k-1} d_{j,b,s,t_j}, 0\}$ equals the stock $v_{k,b,s}$ for Size s in Branch b at the beginning of Sales period k. The number of sold items for the real sales periods is the minimum of the current stock and demand, (4.9) and (4.10) together with the objective function (4.1). At the sellout period $k_{\max}$ all remaining items are sold. Subtracting mark-down costs from the related yield results in the last line of Equation (4.20).

Now we want to deduce the general number of all valid price trajectories. For this purpose we consider a small example.

Example 3 (number of price trajectories)**.** It is $k_{\max} = 4$, i.e. $|K| = 5$, and $p_{\max} = 3$, i.e. $|P| = 4$, and $k_{\text{obs}} = 2$. We depict the sales periods by their indices and encode a mark-down after Period k in Period $k+1$ to the next price index $t_k + 1$ by the symbol $\star$, a mark-down to the after next price index $t_k + 2$ by $\star\star$ and so on. Then these are all valid price trajectories together with their encoding:

price trajectory					encoding							
0	0	0	0	3	0	1	2	3	$\star$	$\star$	$\star$	4
0	0	0	1	3	0	1	2	$\star$	3	$\star$	$\star$	4
0	0	0	2	3	0	1	2	$\star$	$\star$	3	$\star$	4
0	0	1	1	3	0	1	$\star$	2	3	$\star$	$\star$	4
0	0	1	2	3	0	1	$\star$	2	$\star$	3	$\star$	4
0	0	2	2	3	0	1	$\star$	$\star$	2	3	$\star$	4

Using the encoding scheme from Example 3 we can establish the number of all valid price trajectories.

Theorem 2 (number of price trajectories [KKR11b])**.** *The number of all valid price trajectories for Problem 5 is given by*

$$
\binom{k_{\max} - k_{\text{obs}} + p_{\max} - 1}{p_{\max} - 1}. \tag{4.21}
$$

Proof. We encode the feasible price trajectories by inserting $p_{\max}$ symbols for mark-downs, like e.g. $\star$ as in Example 3. In the first k_{obs} observation periods no mark-down is possible, so there is no symbol $\star$ between the related places in the encoding. The last mark-down after Period $k_{\max} - 1$ to the salvage value is determined. So we have to distribute our remaining $p_{\max} - 1$ mark-downs/symbols $\star$ among the remaining $k_{\max} - k_{\text{obs}} + p_{\max} - 1$ places. This yields the claim. □

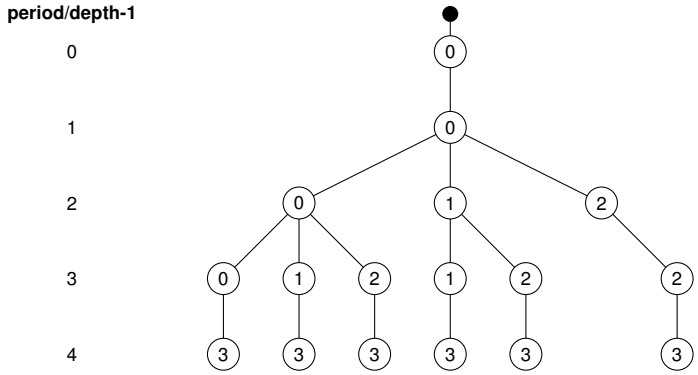

Figure 4.1: POP – enumeration tree

The valid price trajectories can be established by walking through an enumeration tree. The nodes of the tree in depth $k+1$ with $0 \leq k < k_{\max} - 1$ correspond with fixed prices/price indices for the first $k+1$ periods. The leaves at depth $k_{\max}+1$ correspond with valid price trajectories. In every depth $k+1$ with $0 \leq k < k_{\max} - 1$ we consider all extensions with price indices $p = t_{k-1}, \ldots, p_{\max} - 1$. At depth $k_{\max} + 1$ – the sellout period – we have to fix the salvage value, i.e. price index $p_{\max}$.

We consider the enumeration tree for a small example.

Example 4 (enumeration tree for POP)**.** It is $k_{\max} = 4, k_{\text{obs}} = 2$ and $p_{\max} = 3$. In Figure 4.1 the corresponding enumeration tree is depicted. The number inside a node at depth $k+1$ is equivalent to the entry t_k in the price trajectory.

Now we state some corollaries in terms of the enumeration tree which follow from Theorem 2.

Corollary 2 (number of nodes per depth)**.** *The number of nodes in depth $k+1$ with $k_{\text{obs}} \leq k < k_{\max}$ in the enumeration tree amounts to*

$$\binom{k - k_{\text{obs}} + p_{\max}}{p_{\max} - 1}. \tag{4.22}$$

Proof. This Corollary follows easily from Theorem 2 because it is equivalent to consider the enumeration tree for the same data, but $k_{\max} = k + 1$. □

Corollary 3 (number of nodes)**.** *The overall number of nodes in the enumeration tree amounts to*

$$k_{\text{obs}} + \sum_{k=k_{\text{obs}}}^{k_{\max}-1} \binom{k - k_{\text{obs}} + p_{\max}}{p_{\max} - 1} + \binom{k_{\max} - k_{\text{obs}} + p_{\max} - 1}{p_{\max} - 1}. \tag{4.23}$$

Proof. This claim – more precisely the second term – follows from Corollary 2. The first term describes the number of nodes for the observation time k_{obs}. Because for periods k with $k < k_{\text{obs}}$ the starting price has to be maintained there is always one node for these periods. The third term equals the number of nodes at depth $k_{\max} + 1$ which is the number of all valid price trajectories. □

Corollary 4 (number of induced price trajectories)**.** *The number of induced price trajectories by a node at depth $k+1$ with $k_{\text{obs}} \leq k < k_{\max}$ and $t_k = p$ in the enumeration tree is given by*

$$\binom{k_{\max} - k + p_{\max} - p - 2}{p_{\max} - p - 1}. \tag{4.24}$$

Proof. We consider truncated price trajectories ending up with $t_k = p$. The number of induced price trajectories is the number of all valid extensions. So it is equivalent to determine the number of price trajectories for $k_{\max} = k_{\max} - k$, $k_{\text{obs}} = 1$ and $p_{\max} = p_{\max} - p$. □

4.3 Excursus: Dynamic programming

As already mentioned in the introduction a general approach for inventory and pricing problems is dynamic programming. Dynamic programming is based on the Bellman's optimality principle which roughly says that for a dynamic system (Section 4.3.1) every optimal solution consists of optimal partial solutions. This leads to a backwards dynamic programming algorithm which we outline in Section 4.3.2. While this algorithm computes the optimal partial solutions backwards in time for the special case of deterministic problems an algorithm performing forwards in time can be stated. This is outlined in Section 4.3.3. Deterministic dynamic problems can be reduced to shortest path problems and common algorithms for solving shortest path problems can be applied, Section 4.3.4. Sometimes the state-space for dynamic programs is restricted by resource constraints. Thus, in Section 4.3.5 we consider the case of a resource constraint shortest path problem. Because the in Section 4.3.4 proposed methods only regard the length of the partial path they are not suitable for this problem formulation. The explained approach is extended to a *label setting* algorithm. Now each partial path gets a label which includes the length of the path and the still available amounts of the resources. The state space is reduced by comparing the labels. If one label can not lead to a better solution than the other one it is said to be *dominated*. Dominated labels can be excluded from further consideration. In this section we are mainly guided by [Ber05].

4.3.1 General dynamic program

We consider a system of the form

$$x_{k+1} = f_k(x_k, u_k, w_k), k = 0, 1, \ldots, N-1 \tag{4.25}$$

where k is a discrete time index, x_k is a *state* of the system for stage k, u_k is the decision variable or *control* which is selected at time k and w_k is a random parameter. The number of stages is stated by N which is also denoted as *horizon*. With the function f_k the *dynamic of the system* is described.

Additionally we are given a cost function $g_k(x_k, u_k, w_k)$. The total costs are given by

$$g_N(x_N) + \sum_{k=0}^{N-1} g_k(x_k, u_k, w_k). \tag{4.26}$$

$g_N(x_N)$ is also called *terminal cost*.

With S_k we denote the *state space* of x_k. It is $x_k \in S_k$ and analogously we consider a space C_k where $u_k \in C_k$. The *disturbance* w_k is an element from a space D_k. A control is called *admissible* if $u_k \in U(x_k)$ where $U(x_k) \subset C_k$. That means the admissibility of a control at stage k depends on the state x_k in this stage.

The control u_k is selected with the knowledge of the current state x_k. A *policy* or *control law* is a sequence of functions

$$\pi = \{\mu_0, \ldots, \mu_{N-1}\} \tag{4.27}$$

where μ_k maps the state x_k into controls $u_k = \mu_k(x_k)$.

The goal is to minimize the expected cost $J_\pi(x_0)$ of π starting at stage x_0 which is given by

$$J_\pi(x_0) = \mathbb{E}\left\{ g_N(x_N) + \sum_{k=0}^{N-1} g_k(x_k, \mu_k(x_k), w_k) \right\}. \tag{4.28}$$

We only consider admissible policies, that means policies with $\mu_k(x_k) \in U_k(x_k)$ $\forall x_k \in S_k$. The set of all admissible policies is denoted by Π. An *optimal policy* π^* is a policy π that minimizes the costs, that means

$$J_{\pi^*}(x_0) = \min_{\pi \in \Pi} J_\pi(x_0). \tag{4.29}$$

4.3.2 The dynamic programming algorithm

The techniques to solve dynamic programs are based on the *principle of optimality* stated first by Richard Bellman [Bel10].

Definition 2 (principle of optimality)**.** It is $\pi^* = (\mu_0^*, \mu_1^*, \ldots, \mu_{N-1}^*)$ an optimal policy. The truncated policy $(\mu_i^*, \mu_{i+1}^*, \ldots, \mu_{N-1}^*)$ is also optimal for the subproblem to minimize the expected cost

$$\mathbb{E}\left\{ g_N(x_N) + \sum_{k=i}^{N-1} g_k(x_k, \mu_k(x_k), w_k) \right\} \tag{4.30}$$

from Stage i to Stage N.

We now denote with $J_k(x_k)$ the optimal expected cost for starting at Stage k. With the above principle for every initial state x_0 the optimal cost $J^*(x_0)$ equals $J_0(x_0)$ and is given by the last step of the following algorithm. The algorithm proceeds backwards in time from Stage $N-1$ to Stage 0:

$$J_N(x_N) = g_N(x_N), \tag{4.31}$$

$$J_k(x_k) = \min_{u \in U_k(x_k)} \mathbb{E}_{w_k}\{g_k(x_k, u_k, w_k) + J_{k+1}(f_k(x_k, u_k, w_k))\}, k = 0, 1, \ldots, N-1. \tag{4.32}$$

4.3.3 Deterministic Systems

In this Section we focus on deterministic problems. These are problems where the disturbance w_k takes only one value. This may result from the approximation of a

stochastic problem. For deterministic problems for a given policy $(\mu_0, \ldots, \mu_{N-1})$ and the initial state x_0 the future states are predictable by

$$x_{k+1} = f_k(x_k, \mu_k(x_k)), k = 0, 1, \ldots, N-1 \tag{4.33}$$

and the corresponding controls are given by

$$u_k = \mu_k(x_k), k = 0, 1, \ldots, N. \tag{4.34}$$

A deterministic dynamic program can be seen as a shortest path problem in a directed graph with nodes corresponding to stages. The source s corresponds to State x_0 while the sink is an artificial terminal node t that describes the state after adding the terminal costs. The inner nodes correspond to the stages $1, 2, \ldots, N$. There are only arcs between nodes corresponding to state x_k and $x_{k+1}, k = 0, \ldots, N-1$. These arcs describe a transition of the form $x_{k+1} = f_k(x_k, u_k)$. The length of an arc is given by the transition cost $g_k(x_k, u_k)$. Moreover every node related to state x_N is connected with the sink t. The corresponding length of the arc is the terminal cost $g_N(x_N)$.

With this reduction solving a dynamic program to optimality is the same as finding the shortest path in the corresponding graph.

This leads to an forward algorithm for the dynamic program what means that we compute optimal partial solutions beginning from Stage 0 and ending up at Stage N.

With a_{ij}^k we denote the *cost of transition* from Stage k and State $i \in S_k$ to State $j \in S_{k+1}$. The terminal cost of State $i \in S_N$ are denoted by a_{ij}^N.

It is

$$\tilde{J}_N(j) = a_{sj}^0, j \in S_1 \tag{4.35}$$

and

$$\tilde{J}_k(j) = \min_{i \in S_{N-k}} [a_{ij}^{N-k} + \tilde{J}_{k+1}(i)]. \tag{4.36}$$

The optimal cost are given by

$$\tilde{J}_0(j) = \min_{i \in S_N} [a_{ij}^N + \tilde{J}_1(i)]. \tag{4.37}$$

4.3.4 Solving shortest path problems

In the previous section we stated the context of deterministic dynamic programming and shortest path problems and a forward algorithm which can be seen as a general approach to solve shortest path problems. We consider a graph where we want to find the shortest path from a source node s to a sink node t. The length of the path results as the sum of the lengths of the traversed arcs.

The problem can be solved to optimality by a so-called *label correcting* algorithm. The idea is to discover shorter paths from the source s to every other node j and to maintain the length of the shortest path found so far in a variable d_j which is called the *label* of j.

We start from the source s, Step 2 of Algorithm 2, and extend our partial step-by-step to a path ending up at the sink t. For this purpose we consider all possible arcs starting at the end node i of our partial path, Step 5. Whenever a shorter path from the sink to a node j is found, the label is *corrected* in Step 7, i.e. we always consider only the shortest partial path from the source to node j. Because according to the Bellman's optimality principle each path consists of optimal partial paths we will end up with a shortest path from the source s to the sink t.

Algorithm 2 Label correcting

```
1: init d_j = ∞ for all nodes j
2: init OPEN={s}
3: while OPEN≠ ∅ do
4:     choose node i from OPEN
5:     for all childs j of i do
6:         if d_i + a_ij < min{d_j, UPPER} then
7:             d_j = d_i + a_ij
8:             if j ∉ OPEN and j ≠ t then
9:                 place j in OPEN
10:            else
11:                if j = t then
12:                    UPPER=d_i + a_ij
13:                end if
14:            end if
15:        end if
16:    end for
17:    remove i from OPEN
18: end while
```

There are different ways to perform the label correcting algorithm. For example, one could traverse the nodes from the set OPEN in a breadth-first search, also known as Bellman-Ford method with complexity $\mathcal{O}(nm)$ where n is the number of nodes and m the number of arcs. Or one can perform a depth-first search with the same complexity but with the advantage that the amount of needed memory is less. By a best-first search, also denoted by Dijkstra's method the complexity only amounts to $\mathcal{O}(n\log n+m)$. But Dijkstra's algorithm in general works only correctly if there are no negative arc lengths. (Otherwise Bellman's optimality principle may be violated, because with negative arc lengths an optimal path has not necessarily to consist of shortest partial paths.) If the graph contained negative cycles then the label correcting approach would not terminate. Traversing negative cycles would always reduce the length. The Bellman-Ford method can detect negative circles. (In the case of dynamic programming where there are only forwards arcs – from stage k to stage $k+1$ – no cycles can occur.)

The label correcting method can also be extended to a Branch&Bound method where in comparison with the bound UPPER solutions are discarded that have no chance to be optimal.

For further reading about shortest path problems we refer to [CGR93].

4.3.5 Resource constraint shortest path problems and dominance

We now deal with shortest path problems with one or more additional resource constraints. For each resource j an initial stock $R_{init}^{(j)}$ is given. Each arc in the graph, see Section 4.3.3, consumes always an amount of the given resources. Now, a path is only valid if the totally amounts of each resources does not violate the resource restriction – i.e. the sum of the consumed amounts of resource j over all arcs in the path must not exceed $R_{init}^{(j)}$.

For resource constraint shortest path problems the algorithms stated in the last section are not convenient. In a simple shortest path problem the shortest path is always the best. For resource constraint shortest path problems this path might violate the resource constraints.

Thus, we can not exclude longer paths being optimal as it is implied by Step 7 of Algorithm 2. An idea would be to save all possible paths. But according to the problem size this might be inefficient in terms of time and impossible in terms of memory.

Handler and Zang [HZ80] showed that the resource constraint shortest path prob-

lem in general – also in our case where no cycles in the graph appear – is NP hard by reducing the knapsack problem to it. The same is shown by Garey and Johnson [GJ79]. But they reduced the partition problem on it. Irnich and Desaulniers [ID05] among others covered so-called *dominance rules* for resource constraint shortest path problems. The label correcting algorithm above is adapted to a so-called *label setting* algorithm: Not only the shortest path to a node is regarded but also longer paths which cannot be excluded from being optimal.

In our case we are given a resource constraint shortest path problem with n resources. The initial stock for each resource $j = 1, \ldots, n$ is given by $R^{(j)}_{init}$. For each resource $j = 1, \ldots, n$ and each arc from Node k_1 to Node k_2 a consumption or weight $c^{(j)}_{k_1 k_2}$ is given.

With

$$L = (d, R, k) \tag{4.38}$$

we define a *label* for a node. The first element d of the triple is the length of the path from the sink s to the related node k. $R = (r^{(1)}, r^{(2)}, \ldots, r^{(n)})$ is an n-tuple where $r^{(j)}$ is the still available amount of resource j.

We start with the label $L_s = (d_s, R_s, s)$ with

$$d_s = 0, \tag{4.39}$$
$$r^{(j)}_s = R^{(j)}_{init}, j = 1, \ldots, n. \tag{4.40}$$

A label $L_{i_1} = (d_{i_1}, R_{i_1}, k_1)$ is extended to a label L_{i_2} by setting

$$L_{i_2} = (d_{i_2}, R_{i_2}, k_2) \tag{4.41}$$

where

$$d_{i_2} = d_{i_1} + a_{k_1 k_2}, \tag{4.42}$$
$$r^{(j)}_{i_2} = r^{(j)}_{i_1} - c^{(j)}_{k_1 k_2}, j = 1, \ldots, n. \tag{4.43}$$

A label L_{n_1} – which stands for a partial path starting at the sink s and ending at node k_1 – is said to be *dominating* over a label L_{n_2} ending at the same node if $d_{n_1} < d_{n_2}$ and $r^{(j)}_{n_1} \geq r^{(j)}_{n_2}$ for all $j = 1, \ldots, n$. Because of the higher amount of the resource and both labels ending up at the same node we can extend the path described by label L_{n_1} in each way we can extend L_{n_2}. And, because $d_{n_1} < d_{n_2}$ each path containing the to L_{n_2} related partial path cannot be shorter than each path containing the to L_{n_1} related partial path. Thus, we can exclude the label L_{n_2} from further consideration. We say, L_{n_2} *is dominated* by L_{n_1}.

Many further references regarding dominance and dominance rules can be found in literature. We just mention some few of them exemplary. A definition of dominance was given by Manne [Man58] already in the year 1958. In [JC11] dominance rules in combinatorial optimization and their characteristics are generally defined and studied. Fischetti and Salvagnin presented a dominance procedure for general mixed-integer linear programs in [FT88]. General results for applying dominance in Branch&Bound algorithms are presented in [Iba77].

4.4 Dynamic generation of mark-down strategies

In the POP$^{\hat{e}}$ we deal with fixed supply for each branch and size, Constraint (4.7). We apply a dynamic programming approach, see the last section, to solve Problem 5. In

the formulation of the POP$^{\hat{e}}$ we treated demand as a predetermined size. Thus, we can see our formulation of the price optimization as a deterministic problem. We have to maintain a resource restriction for each branch and size: the number of items may not exceed the supply. So we are able to treat POP$^{\hat{e}}$ as a resource constraint shortest path problem where a period equals a stage. We start from Period 0 and end up at Period $k_{\max}$. We will describe how to find the optimal mark-down policy by dynamic generation of the price trajectories.

At first we will define our labels which describe revenue, remaining stock per branch and size and the set prices. A label is denoted as a *complete mark-down strategy* or a *partial mark-down strategy*.

Definition 3 (complete mark-down strategy)**.** A complete mark-down strategy P is defined as a tupel $P = (a, t)$. The $k_{\max}$+1-tuple $t = (t_1, \ldots, t_{k_{\max}})$ is the *price trajectory* of P, see Definition 1 and a the related revenue. We also call a the *revenue* of P in this context.

A mark-down strategy arises by extending a *partial mark-down strategy*:

Definition 4 (partial mark-down strategy)**.** A partial mark-down strategy $\tilde{P}$ is defined as a tupel $\tilde{P} = (\tilde{a}, \tilde{t}, \tilde{r})$ where $\tilde{a} \in \mathbb{R}$ is called *partial revenue* of $\tilde{P}$. The k+1-tuple $\tilde{t} = (t_0, \ldots, t_k)$ are the ordered price indices assigned to the sales periods $0 \ldots, k$ called *partial price trajectory* of $\tilde{P}$. The *stock* of $\tilde{P}$ is given by $\tilde{r} \in \mathbb{R}^{|B|\times|S|}$, where $\tilde{r}_{b,s}$ is the stock for Branch b and Size s after Period k. It is $0 \leq k < k_{\max}$.

Definition 5 (valid extension of a partial mark-down strategy)**.** Given is a partial mark-down strategy $\tilde{P}^{(1)} = \left(\tilde{a}^{(1)}, \tilde{t}^{(1)}, \tilde{r}^{(1)}\right)$ with $\tilde{t}^{(1)} = \left(t_0^{(1)}, \ldots, t_k^{(1)}\right)$. We consider the partial mark-down strategy $\tilde{P}^{(2)} = \left(\tilde{a}^{(2)}, \tilde{t}^{(2)}, \tilde{r}^{(2)}\right)$ with $\tilde{t}^{(2)} = \left(t_0^{(2)}, \ldots, t_{k+1}^{(2)}\right)$ and $k + 1 < k_{\max}$. We introduce the parameter $\hat{\beta}_{k+1} \in \{0, 1\}$ that takes value one if and only if $t_k^{(2)} \neq t_{k+1}^{(2)}$. The partial mark-down strategy $\tilde{P}^{(2)}$ is called a valid extension of $\tilde{P}^{(1)}$ if the following conditions hold:

$$t_i^{(2)} = 0 \quad \forall 0 \leq i < k_{\text{obs}}, \tag{4.44}$$

$$t_{k+1}^{(2)} \geq t_k^{(1)}, \tag{4.45}$$

$$\left(t_0^{(2)}, \ldots, t_k^{(2)}\right) = \left(t_0^{(1)}, \ldots, t_k^{(1)}\right), \tag{4.46}$$

$$\begin{aligned} \tilde{a}^{(2)} \quad = \quad & \tilde{a}^{(1)} + \exp(-\rho(k+1)) \\ & \cdot \Bigg(\pi_{t_{k+1}^{(2)}} \cdot \sum_{b\in B} \sum_{s\in S} \min \left\{ \tilde{r}_{b,s}^{(1)}, d_{k+1,b,s,t_{k+1}^{(2)}} \right\} \\ & - \hat{\beta}_{k+1} \cdot \left(\mu_f + \mu_v \cdot \sum_{b\in B} \sum_{s\in S} \tilde{r}_{b,s}^{(1)} \right) \Bigg), \end{aligned} \tag{4.47}$$

$$\tilde{r}_{b,s}^{(2)} = \max \left\{ \tilde{r}_{b,s}^{(1)} - d_{k+1,b,s,t_{k+1}^{(2)}}, 0 \right\} \quad \forall b \in B, s \in S. \tag{4.48}$$

In terms of dynamic programming, an extension of a partial mark-down strategy with a price trajectory of length k equals a transition from Stage – in our case Period – k to $k+1$. The dynamic of the system is given by (4.47) and equals the revenue at the end of Period k. Our control is given by the price index t_k we assign to Period k and the truncated policy is described by the partial price trajectory.

Equation (4.44) analogously to Equation (4.3) guarantees that the starting price is maintained for the first k_{obs} periods. The prices have to be non-increasing, i.e. the price indices non-decreasing; see also Constraint (4.5). This is ensured by Inequality (4.45). Additionally sells must not exceed the initial stock. That means we are given resource constraints for each branch and size. The stock is updated by Equation (4.48).

We defined partial mark-down strategies only for sales periods $k < k_{\max}$. To extend a partial mark-down strategy to a complete mark-down strategy we have to add salvage value and to subtract mark-down costs for all remaining items as terminal costs.

Definition 6 (valid extension to a complete mark-down strategy)**.** Given is a partial mark-down strategy $\tilde{P}^{(1)} = \left(\tilde{a}^{(1)}, \tilde{t}^{(1)}, \tilde{r}^{(1)}\right)$ with $\tilde{t}^{(1)} = \left(t_0^{(1)}, \ldots, t_{k_{\max}-1}^{(1)}\right)$. The mark-down strategy $P = (a, t)$ with $t = (t_0, \ldots, t_{k_{\max}})$ is called an valid extension of $\tilde{P}$ if the following conditions hold:

$$(t_0, \ldots, t_{k_{\max}-1}) = \left(t_0^{(1)}, \ldots, t_{k_{\max}-1}^{(1)}\right), \tag{4.49}$$

$$a = \tilde{a}^{(1)} + \exp(-\rho k_{\max}) \cdot (\pi_{p_{\max}} - q_{k_{\max}} \mu_v) \cdot \sum_{b \in B} \sum_{s \in S} \tilde{r}_{b,s}^{(1)}. \tag{4.50}$$

So every mark-down strategy can be established by extensions of a partial mark-down strategy. Starting with $\tilde{P}^s = (\tilde{a}^s, \tilde{t}^s, \tilde{r}^s)$ where $\tilde{a}^s = 0$, $\tilde{t}^s = ()$ and the stock being $\tilde{r}_{b,s} = I_{b,s}\ \forall b \in B, \forall s \in S$ we can obtain every valid mark-down strategy from $\tilde{P}^s$ by $k_{\max} + 1$ extensions.

4.5 Pruning the enumeration tree – dominating partial mark-down strategies

Dominance rules allow us to reduce the state space by excluding mark-down strategies from further consideration since they cannot lead to optimality.

We develop dominance rules specifically for the Price Optimization Problem.

Definition 7 (dominating partial mark-down strategy)**.** Consider the two partial mark-down strategies $\tilde{P}^{(1)} = \left(\tilde{a}^{(1)}, \tilde{t}^{(1)}, \tilde{r}^{(1)}\right)$ and $\tilde{P}^{(2)} = \left(\tilde{a}^{(2)}, \tilde{t}^{(2)}, \tilde{r}^{(2)}\right)$. The price trajectories are given by $t^{(1)} = \left(t_0^{(1)}, \ldots, t_{k^{(1)}}^{(1)}\right)$ and $t^{(2)} = \left(t_0^{(2)}, \ldots, t_{k^{(2)}}^{(2)}\right)$. The parameter $\tau \in \{0, 1\}$ takes value one if and only if $t_{k^{(1)}}^{(1)} \neq t_{k^{(2)}}^{(2)}$. We say that $\tilde{P}^{(1)}$ *dominates* $\tilde{P}^{(2)}$ if

$$k^{(1)} \leq k^{(2)}, \tag{4.51}$$

$$t_{k^{(1)}}^{(1)} \leq t_{k^{(2)}}^{(2)}, \tag{4.52}$$

$$\begin{aligned}
&\tilde{a}^{(1)} - \tau\exp\left(-\rho\left(k^{(2)}+1\right)\right)\left(\mu_f + \mu_v \cdot \sum_{b\in B}\sum_{s\in S}\tilde{r}^{(1)}_{b,s}\right) \\
&- \sum_{k=k^{(2)}+1+\tau}^{k^{(2)}+1+\tau+\min\left\{k_{\max}-k^{(2)}-\tau-1,\, p_{\max}-1-t^{(2)}_{k^{(2)}}\right\}} \exp(-\rho k)\mu_v \sum_{b\in B}\sum_{s\in S}\tilde{r}^{(1)}_{b,s} \\
&-\exp(-\rho k_{\max})q_{k_{\max}}\mu_v \sum_{b\in B}\sum_{s\in S}\tilde{r}^{(1)}_{b,s} > \tilde{a}^{(2)}.
\end{aligned} \tag{4.53}$$

We call $\tilde{P}^{(1)}$ *dominating* in terms of $\tilde{P}^{(2)}$ and $\tilde{P}^{(2)}$ *dominated by* $\tilde{P}^{(1)}$.

There is no need for a further consideration of a dominated mark-down strategy.

Theorem 3 (pruning dominated mark-down strategies)**.** *With $\tilde{P}^{(1)} = \left(\tilde{a}^{(1)}, \tilde{t}^{(1)}, \tilde{r}^{(1)}\right)$ with $\tilde{t}^{(1)} = \left(t^{(1)}_0, \ldots, t^{(1)}_{k^{(1)}}\right)$ and $\tilde{P}^{(2)} = \left(\tilde{a}^{(2)}, \tilde{t}^{(2)}, \tilde{r}^{(2)}\right)$ with $\tilde{t}^{(2)} = \left(t^{(2)}_0, \ldots, t^{(2)}_{k^{(2)}}\right)$ we are given two partial mark-down strategies. The partial mark-down strategy $\tilde{P}^{(1)}$ dominates $\tilde{P}^{(2)}$. Then every valid extension of $\tilde{P}^{(2)}$ applied on $\tilde{P}^{(1)}$ would yield a higher revenue for $\tilde{P}^{(1)}$.*

To prove Theorem 3 we will start with the following Lemma:

Lemma 1 (additional mark-down costs of extensions)**.** *Let $\tilde{P}^{ext} = \left(\tilde{a}^{ext}, \tilde{t}^{ext}, \tilde{r}^{ext}\right)$ with $\tilde{t}^{ext} = (t^{ext}_0, \ldots, t^{ext}_k)$ and $\tilde{P}^{(2)} = \left(\tilde{a}^{(2)}, \tilde{t}^{(2)}, \tilde{r}^{(2)}\right)$ with $\tilde{t}^{(2)} = \left(t^{(2)}_0, \ldots, t^{(2)}_k\right)$ be two partial mark-down strategies with price trajectories of the same length and $t^{ext}_k \leq t^{(2)}_k$.*

Let $\tau \in \{0, 1\}$ be as in Definition 7.

Then for all possible complete mark-down strategies arising by extensions of $\tilde{P}^{(2)}$ the additional overall mark-down costs which arise by applying the same extension to $\tilde{P}^{ext}$ are bounded above by

$$\begin{aligned}
&\tau\exp(-\rho(k+1))\left(\mu_f + \mu_v \cdot \sum_{b\in B}\sum_{s\in S}\tilde{r}^{ext}_{b,s}\right) \\
&+ \sum_{j=k+1+\tau}^{k+1+\tau+\min\left\{k_{\max}-k-\tau-1,\, p_{\max}-1-t^{(2)}_k\right\}} \exp(-\rho j)\mu_v \sum_{b\in B}\sum_{s\in S}\tilde{r}^{ext}_{b,s} \\
&+\exp(-\rho k_{\max})q_{k_{\max}}\mu_v \sum_{b\in B}\sum_{s\in S}\tilde{r}^{ext}_{b,s}
\end{aligned} \tag{4.54}$$

Proof. We consider two cases: In the first case we assume that the price trajectories of both partial mark-down strategies $\tilde{P}^{ext}$ and $\tilde{P}^{(2)}$ end up with the same price index, in the second case that they do not.

Case 1: $t^{ext}_k = t^{(2)}_k$

The price trajectories $\tilde{t}^{ext}$ and $t^{(2)}$ of both partial mark-down strategies $\tilde{P}^{ext}$ and $\tilde{P}^{(2)}$ end up with the same price index. This means all valid extensions of $\tilde{P}^{(2)}$ are also valid for $\tilde{P}^{ext}$. The maximum number of additional mark-downs for a partial mark-down strategy with a price trajectory of length k and last price t_k amounts to

$$\min\{k_{\max} - k - 1, p_{\max} - 1 - t_k\}. \tag{4.55}$$

The first term describes the bounding by the remaining periods to obtain a complete mark-down strategy, the second one the bounding by the remaining price steps.

The additional overall fixed mark-down costs do not differ if we extend $\tilde{P}^{ext}$ and $\tilde{P}^{(2)}$ the same way, so we can neglect it. The variable mark-down costs depend on the stock at the beginning of the corresponding period. Because of the discounting and the non-increasing stock the sooner a mark-down happens the higher costs for mark-downs are. With taking into account that the demand for the subsequent periods can take value zero for all branches and sizes – that means the remaining items may have to be priced after each further mark-down decision including the $q_{k_{\max}}$ mark-downs during sellout – we get

$$\sum_{j=k+1}^{k+1+\min\{k_{\max}-k-1,p_{\max}-1-t_k^{(2)}\}} \exp(-\rho j)\mu_v \sum_{b\in B}\sum_{s\in S}\tilde{r}_{b,s}^{ext} + \exp(-\rho k_{\max})q_{k_{\max}}\mu_v \sum_{b\in B}\sum_{s\in S}\tilde{r}_{b,s}^{ext}$$

as an upper bound for the additional mark-down costs for all valid extensions of $\tilde{P}^{ext}$.

Case 2: $t_k^{ext} < t_k^{(2)}$

In this case not all valid extensions of $\tilde{P}^{ext}$ are also valid for $\tilde{P}^{(2)}$. If we consider the valid extensions of $\tilde{P}^{(2)}$ with $t_{k+1}^{(2)} = t_k^{(2)}$ – i.e. the price index for the next period is the same as for period k – we have to take into account that additional mark-down costs can arise if we extend $\tilde{P}^{ext}$ the same way. The maximal additional mark-down costs arise if $\tilde{P}^{ext}$ immediately in period $k^{ext}+1$ is extended by $t_k^{(2)}$ (through the non-increasing stock and the discounting). Additionally this time we also have to take additional fixed mark-down costs for period $k^{ext}+1$ into account. After extending $\tilde{P}^{ext}$ and $\tilde{P}^{(2)}$ by $t_k^{(2)}$ we are in a similar situation as in the first case: The last indices of the both price trajectories are the same, but now they are of length $k+1$. So we can bound the additional mark-down costs by

$$\exp(-\rho(k+1))\left(\mu_f + \mu_v \cdot \sum_{b\in B}\sum_{s\in S} r_{b,s}^{ext}\right) + \sum_{j=k+2}^{k+2+\min\{k_{\max}-k-2,p_{\max}-1-t_k^{(2)}\}} \exp(-\rho j)\mu_v \sum_{b\in B}\sum_{s\in S}\tilde{r}_{b,s}^{ext} + \exp(-\rho k_{\max})q_{k_{\max}}\mu_v \sum_{b\in B}\sum_{s\in S}\tilde{r}_{b,s}^{ext}.$$

For extensions with $t_{k+1}^{(2)} > t_k^{(2)}$, we can neglect the fixed costs for extending $\tilde{P}^{ext}$ in Period $k+1$ by t_{k+1} because they would also arise if $\tilde{P}^{(2)}$ is extended the same way. So the upper bound for the additional mark-down costs is also valid in this case.

With including the parameter τ that indicates if the last prices for $\tilde{P}^{ext}$ and $\tilde{P}^{(2)}$ are different, we obtain bound (4.54). □

Now we prove Theorem 3.

Proof of Theorem 3. We split up the proof in two cases. At first we assume that both, the fixed and the variable mark-down cost, take value zero. Then we will go into case

that the mark-down costs take arbitrary positive values.

Case 1: $\mu_f = \mu_v = 0$

According to Definition 7 it is

- $k^{(1)} \leq k^{(2)}$,
- $t^{(1)}_{k^{(1)}} \leq t^{(2)}_{k^{(2)}}$,
- $\tilde{a}^{(1)} > \tilde{a}^{(2)}$.

Consider the partial mark-down strategy $\tilde{P}^{ext} = \left(\tilde{a}^{ext}, \tilde{t}^{ext}, \tilde{r}^{ext}\right)$ with price trajectory $\tilde{t}^{ext} = \left(t^{(1)}_0, \ldots, t^{(1)}_{k^{(1)}}, t^{ext}_{k^{(1)}+1}, t^{ext}_{k^{(1)}+2}, \ldots, t^{ext}_{k^{(2)}}\right)$ where the price does not change after Period $k^{(1)}$, i.e. $t_{k^{(1)}} = t^{ext}_{k^{(1)}+1} = t^{ext}_{k^{(1)}+2} = \ldots = t^{ext}_{k^{(2)}}$.

Because $t^{ext}_{k^{(2)}} \leq t^{(2)}_{k^{(2)}}$ by assumption it is possible to extend $\tilde{P}^{ext}$ in every way as $\tilde{P}^{(2)}$ can be extended. In this consideration the current stock does not play a role. Because $\mu_v = \mu_f = 0$ the revenue per period is completely given by the yield per period. In this case a higher stock is never unfavorable in terms of revenue. Due to the non-increasing pricing selling an item earlier is always better. (If $\tilde{r}^{ext}_{b,s} < \tilde{r}^{(2)}_{b,s}$ than we will not be able to get a higher revenue for the non-sold $\tilde{r}^{(2)}_{b,s}$ items by extending $\tilde{P}^{(2)}$ because $t^{(2)}_{k^{(2)}} \geq t^{ext}_{k^{(2)}}$. If $\tilde{r}^{ext}_{b,s} > \tilde{r}^{(2)}_{b,s}$ then – because we extend both price trajectories the same way – we can meet more demand for the extensions of $\tilde{P}^{ext}$ and therefore obtain a higher revenue than for the same extension of $\tilde{P}^{(2)}$.) It is $\tilde{a}^{ext} \geq \tilde{a}^{(1)} > \tilde{a}^{(2)}$ because no negative costs arise. If we extend $\tilde{P}^{ext}$ and $\tilde{P}^{(2)}$ the same way we therefore get a greater revenue for the extension of $\tilde{P}^{ext}$.

It follows that for every complete mark-down strategy resulting from extensions of $\tilde{P}^{(2)}$ there is a mark-down strategy resulting from $\tilde{P}^{(1)}$ with greater revenue.

Case 2:$\mu_f \neq 0$ and/or $\mu_v \neq 0$

We now have to regard that also negative costs can occur in a sales period depending on if there is a mark-down and the current stock. The requirements are tighter than in the previous case, Definition 7, (4.53). With Lemma 1 the claim follows from the first case.

□

Remark 4. *With Definition 7 and Theorem 3 we defined dominance rules that allow us to exclude partial mark-down strategies from further consideration. One could think about tightening Condition (4.53) in Definition 7. Now we obtain the bound by the implicit assumption that the additional mark-down costs have to be applied for all non-sold items in terms of the partial mark-down strategy* $\tilde{P}^{(1)}$*. But actually in the first case of Lemma 1 we would only have to consider additional mark-down costs for* $\max\left\{\tilde{r}^{ext}_{b,s} - \tilde{r}^{(2)}_{b,s}, 0\right\}$ *items for each branch* b *and size* s*. Replacing* $\tilde{r}^{ext}_{b,s}$ *in the second and third line of (4.53) by this term would also yield a valid bound. While in the implementation (Algorithm 4) our label additionally contains the overall stock* $\sum_{b\in B}\sum_{s\in S}\tilde{r}_{b,s}$*, tightening (4.53) in Definition 7 as described would require traversing all branches and sizes for each dominance check. So far we can not make a point about how a tightened dominance rule for our instances would affect our results in terms of the computation time.*

Replacing $\sum_{b\in B}\sum_{s\in S}\tilde{r}^{ext}_{b,s}$ by $\sum_{b\in B}\sum_{s\in S}\tilde{r}^{ext}_{b,s} - \sum_{b\in B}\sum_{s\in S}\tilde{r}^{2}_{b,s}$ in (4.53) in general does not lead to an upper bound for the additional mark-down costs.

Remark 5. *One could think about computing the number of items which in terms of $\tilde{P}^{ext}$ in contrast to $\tilde{P}^{(2)}$ additionally may have to be marked down by add the related additional stock for all branches and sizes, i.e. $\sum_{b\in B}\sum_{s\in S}\tilde{r}^{ext}_{b,s} - \sum_{b\in B}\sum_{s\in S}\tilde{r}^{2}_{b,s}$. But in general this is not correct. For simplicity we take a look on the case of only one branch with index $b = 1$ with two sizes $S = \{S, L\}$ and $\tilde{r}^{ext}_{1,S} = 1$, $\tilde{r}^{ext}_{1,L} = 2$, $\tilde{r}^{(2)}_{1,S} = 2$, $\tilde{r}^{(2)}_{1,L} = 0$. Summing up and subtracting the sums as in the formula above would yield to additional mark-down costs for 1 item for $\tilde{P}^{ext}$ for all following periods. But this is not a correct assumption. If extending $\tilde{P}^{ext}$ and $\tilde{P}^{(2)}$ the same way led to a demand of 2 items for Size S and 0 items for Size L for the next period, the remaining stocks after this period would be $\tilde{r}^{ext}_{1,S} = 0$, $\tilde{r}^{ext}_{1,L} = 2$, $\tilde{r}^{(2)}_{1,S} = \tilde{r}^{(2)}_{1,L} = 0$. This would yield to mark-down costs for two additional items for the current period. That means summarizing the remaining stocks and compare them is not necessarily correct to estimate the additional mark-down costs. But if we would assume that revenues for selling an item would always exceed the variable costs for mark-downs adding up the stock over all branches and sizes would yield a valid bound anyway. But in general this is not the case, as the example shows.*

4.6 Implementation

We implemented the dynamic generation of price trajectories as a Branch&Bound algorithm in a depth-first-search, see Algorithm 3.

The actual depth-first-search is outlined in Algorithm 4. We extend our partial mark-down strategy as described in Definition 5 and 6 to partial and complete mark-down strategies. For mark-down strategies with a price trajectory ending up with a period with index smaller than $k_{\max} - 1$ we perform a dominance check, see Algorithm 5. (We do not run the dominance check for partial mark-down strategies with $k_{\max} - 1$ periods because the extension to the complete mark-down strategy as per Definition 6 is by regarding the overall stock $\sum_{b\in B}\sum_{s\in S}\tilde{r}_{b,s}$ in the labels done in $\mathcal{O}(1)$ – multiplying stock with salvage value minus variable mark-down costs yields the additional revenue for period $k_{\max}$.)

For checking dominance we initialize a $(k_{\max}-1)\times(p_{\max}-1)$–dimensional array with value $-\infty$ at the beginning of Algorithm 3, values $a^{\text{bound}}_{k,p}$. This array contains for each period k with $k < k_{\max} - 1$ and each price index p with $p < p_{\max}$ a lower bound for the revenue of all extensions of mark-down strategies which start with a price index greater or equal than p and end up at period k. We perform the dominance check by comparing the current revenue of the considered partial mark-down strategy $\tilde{P} = (\tilde{a}, \tilde{t}, \tilde{r})$ with $\tilde{t} = (t_0, \ldots, t_k)$ with the current value of a^{bound}_{k,t_k}. If $\tilde{a} \leq a^{\text{bound}}_{k,t_k}$ the check for dominance is successful and we can prune the current branch of the enumeration tree. Otherwise we try to update the values $a^{\text{bound}}_{k,p}$ with $p \geq t_k$. For this purpose we compute the maximal additional mark-down costs that can arise by extending $\tilde{P}$ with all valid extensions starting with a price index greater or equal than t_k. If the revenue of $\tilde{P}$ less the maximal mark-down costs is greater than the current value of $a^{\text{bound}}_{k,p}$, then $a^{\text{bound}}_{k,p}$ is updated.

The optimal mark-down strategy P^* is updated whenever a complete mark-down strategy with higher revenue is found.

4.7 An accompanying example

In order to illustrate basic ideas and algorithms we introduce a small (manageable) instance for ISPO on which we will draw on in the remaining course of the thesis.

We consider just two branches and sizes. The selling time amounts to five periods. There are four different prices, salvage value included. This implies at most two possible markdowns during the sales process. Concisely, our data is as follows:

- $|B| = 2, B = \{1, 2\}$
- $|S| = 2, S = \{\mathrm{S}, \mathrm{L}\}$
- $k_{\max} = 4$
- $k_{\mathrm{obs}} = 2$
- $p_{\max} = 4, \pi_0 = 10.99, \pi_1 = 5.99, \pi_2 = 1.99, \pi_3 = 0.99$
- $\mu_f = 1, \mu_v = 0.10$
- $\rho = 0.01$
- $q_{k_{\max}} = 2$
- $\mathrm{ap} = 0.5$
- $E = \{\text{low seller}, \text{normal seller}, \text{high seller}\}$ with $\mathrm{Prob}(\text{low seller}) = 0.2, \mathrm{Prob}(\text{normal seller}) = 0.5, \mathrm{Prob}(\text{high seller}) = 0.3$.
- $\kappa = 1$
- $\delta_1 = 1.5$
- $\mathrm{pcost} = 0.01$
- $\underline{I} = 5, \overline{I} = 10$
- $v^{\min} = 1, v^{\max} = 1, vl^{\min} = 2, vl^{\max} = 3$

In the subsequent example we will assume a fixed supply per branch $b \in B$ and size $s \in S$ which is given by the following table:

b/s	S	L
1	5	5
2	3	8

In the following tables we state the demands per price index p and period k for every pair of branches and sizes (b, s) for the "normal seller" scenario.

$(1, \mathrm{S})$

k/p	0	1	2
0	2	0	0
1	1.5	0	0
2	1	1.5	2
3	0.5	1	1.5

$(1, \mathrm{L})$

k/p	0	1	2
0	3	0	0
1	2	0	0
2	1	2	3
3	0.7	0.8	0.9

$(2, \mathsf{S})$

k/p	0	1	2
0	1	0	0
1	0.9	0	0
2	0.8	0.9	1
3	0.7	0.8	0.9

$(2, \mathsf{L})$

k/p	0	1	2
0	2	0	0
1	1	0	0
2	0.9	1.2	1.5
3	0.5	1	1.4

The demands for the other scenarios are given by

$$d^{\text{low seller}}_{k,b,s,p} = 0.5 \cdot d^{\text{normal seller}}_{k,b,s,p} \quad \forall k \in \{0,1,2,3\}, s \in \{\mathrm{S},\mathrm{L}\}, b \in \{1,2\}, p \in \{0,1,2\}, \tag{4.56}$$

$$d^{\text{high seller}}_{k,b,s,p} = 1.3 \cdot d^{\text{normal seller}}_{k,b,s,p} \quad \forall k \in \{0,1,2,3\}, s \in \{\mathrm{S},\mathrm{L}\}, b \in \{1,2\}, p \in \{0,1,2\}. \tag{4.57}$$

4.8 POP-DYN applied on the accompanying example

We depict the basic ideas of our Algorithm 3, POP-DYN, with help of the example from the last section.

Example 5 (solving POP$^{\hat{e}}$ by Algorithm 3)**.** All valid (partial) mark-down strategies for the example from Section 4.7 are pictured in the enumeration tree in Figure 4.2. Revenue and stock can be found beside the nodes. Nodes which can be pruned by dominance are colored gray.

We just consider the scenario "normal seller". We start by extending the partial mark-down strategy $\tilde{P} = (\tilde{a}, \tilde{t}, \tilde{r})$ with $\tilde{a} = 0$, $\tilde{t} = ()$ and $\tilde{r}_{b,s} = I_{b,s}, \forall s \in S, \forall b \in B$. Because $k_{\text{obs}} = 2$ we have to maintain the starting price with Price index 0 in the two first periods.

For Branch 1 and Size S the demand in Period 0 amounts to 2, the current stock is 5. That means all the demand for this branch and size is met and we obtain a revenue of $\exp(-0 \cdot 0.10) \cdot 2 \cdot 10.99 = 21.98$ for this branch and size. The revenue for Branch 1 and Size L is given by $\exp(-0 \cdot 0.01) \cdot 3 \cdot 10.99 = 32.97$. For Branch 2 we earn 10.99 and for Size L 21.98. There is no mark-down in this period. Hence, the revenue for all branches and sizes is the sum of the single revenues, namely 87.92.

We get to Period 1. For Branch 1 and Size S there are 3 items left, for Size L 2. In Branch 2 there are still 2 items of Size S and 6 items of Size L available. Thus, all demand of this period can be met and our updated revenue at the end of Period 1 is $87.92 + \exp(-1 \cdot 0.01) \cdot 10.99 \cdot (1.5 + 2 + 0.9 + 1) = 146.68$.

Because we traverse the enumeration tree by depth-first-search we first consider the extension of the current partial mark-down strategy with Price Index 0. In Branch 1 the demand for Size S can be met, Size L is sold out. The demands for both sizes in Branch 2 can be fully met. So the revenue for the current partial mark-down strategy amounts to $146.68 + \exp(-2 \cdot 0.01) \cdot 10.99 \cdot (1 + 0 + 0.8 + 0.9) = 175.76$.

Now we perform the dominance check. Because we stand in the first branch of the enumeration tree pruning is not yet possible. But we are able to update our bounds $a^{\text{bound}}_{2,p}, p = 0, 1, 2$ which still take value $-\infty$. For the current period $k = 2$ and price index p there could be only one additional mark-down in the extension (see 4.55) and with the maximum mark-down costs that can arise in Period $k_{\max}$ we get

$$\begin{aligned} a_\mu &= \exp(-3 \cdot 0.01) \cdot (0.5 + 0 + 0.3 + 4.1) \cdot 0.1 \\ &+ \exp(-4 \cdot 0.01) \cdot ((0.5 + 0 + 0.3 + 4.1) \cdot 2 \cdot 0.1 + 2 \cdot 1) = 3.34. \end{aligned}$$

So the updated bound amounts to $a_{2,0}^{\text{bound}} = 175.76 - 3.34 = 172.42$.

For $p = 1$ we have to regard that in comparison to a mark-down strategy of the same length with the current last price index 0 an additional mark-down may be possible, that means we update our bound to

$$\begin{aligned} a_{2,1}^{\text{bound}} &= 175.76 - \exp(-3 \cdot 0.01) \cdot (0.1 \cdot (0.5 + 0 + 0.3 + 4.1) + 1) \\ &+ \exp(-4 \cdot 0.01) \cdot ((0.5 + 0 + 0.3 + 4.1) \cdot 2 \cdot 0.1 + 2 \cdot 1) = 171.45. \end{aligned}$$

All new bounds for Period 2 are outlined in the following table:

$\mathbf{p}$	0	1	2
$\mathbf{a}_{\mathbf{2,p}}^{\text{bound}}$	172.42	171.45	171.45

At Period 3 we extend the current partial mark-down strategy again with Price Index 0. In Branch 1 the demand of Size S can be met while Size L is as always sold out. In Branch 2 we meet the demand only partly and sell all remaining items of Size S. The demand for Size L can be fully met. Therefore the revenue is given by $175.76 + \exp(-3 \cdot 0.01) \cdot 10.99 \cdot (0.5 + 0 + 0.3 + 0.5) = 189.63$.

Now we come to extend the partial mark-down strategy to a complete one. For every remaining item on the one hand we earn the salvage value on the other hand we have to pay two times variable mark-down cost. We obtain a complete strategy with revenue $189.63 + \exp(-4 \cdot 0.01) \cdot ((0.99 - 2 \cdot 0.1) \cdot (0 + 0 + 0 + 3.6) - 2 \cdot 1) = 190.44$. We set $P^* = (190.44, (0, 0, 0, 0, 3))$.

The next step in the depth-first-search is to extend the partial mark-down strategy with partial price trajectory $(0, 0, 0)$ by Price index 1. This implies a mark-down at the beginning of Period 3 and together with the yield earned by the sales the corresponding revenue amounts to $175.76 + \exp(-3 \cdot 0.01) \cdot (5.99 \cdot (0.5 + 0 + 0.3 + 1) - 2 \cdot 1 - 2 \cdot 0.1 \cdot (0.5 + 0 + 0.3 + 4.1)) = 184.78$.

Extending the current partial mark-down strategy to a complete one yields a revenue of 185.21.

We have to continue the algorithm without the possibility to prune nodes until our partial mark-down strategy will end up in Period 2 with Price Index 1. The revenue for this strategy amounts to 166.09. Because $166.09 < 171.45 = a_{2,1}^{\text{bound}}$ the current branch of the tree can be pruned. This is the same with Period 2 and Price Index 2.

We end up with the optimal mark-down strategy $P^* = (190.44, (0, 0, 0, 0, 3))$.

Algorithm 3 POP-DYN

Require: complete data of an instance of the POP$^{\hat{e}}$
Ensure: optimal mark-down strategy $P^* = (a^*, t^*)$
1: init $a_{k,p}^{\text{bound}} = -\infty, \forall k < k_{\max} - 1, \forall p < p_{\max}$,
init $\tilde{a} = 0$,$\tilde{t} = ()$, $\tilde{r}_{b,s} = I_{b,s}$ $\forall b \in B, \forall s \in S$,
init $P^* = (a^*, t^*)$ with $a^* = -\infty$ and t^* uninitialised
2: set $\tilde{P} = (\tilde{a}, \tilde{t}, \tilde{r})$
3: POP-DFS($\tilde{P}$)
4: return P^*

Algorithm 4 POP-DFS

Require: a partial mark-down strategy $\tilde{P} = (\tilde{a}, \tilde{t}, \tilde{r})$ with $\tilde{t} = (t_0, \ldots, t_k)$
Ensure: valid non-dominated extensions of $\tilde{P}$

```
if k < k_max − 1 then
    if POP-DOM(P̃)=true then
        return
    end if
    for all valid extensions P̃^ext of P̃ (see Definition 5) do
        POP-DFS(P̃^ext)
    end for
else
    extend P̃ to obtain the complete mark-down strategy P = (a, t) (see Definition 6)
    if a > a* then
        P* = P
    end if
end if
```

Algorithm 5 POP-DOM

Require: a partial mark-down strategy $\tilde{P} = (\tilde{a}, t, \tilde{r})$ with $\tilde{t} = (t_0, \ldots, t_k)$,
for each price index $p : p < p_{\max}$ a bound $a_{k,p}^{\text{bound}}$
Ensure: possibly updated bound $a_{k,p}^{\text{bound}}$ for $t_k \leq p < p_{\max}$,
$\tilde{P}$ dominated? true or false

```
if ã ≤ a^bound_{k,t_k} then
    return true
else
    for all p : p ≥ t_k, p < p_max do
        compute the maximal additional mark-down costs ã_μ for extending P̃ by all valid extensions starting with the price index p (see Lemma 1)
        if ã − ã_μ > a^bound_{k,p} then
            a^bound_{k,p} = ã − ã_μ
        end if
    end for
    return false
end if
```

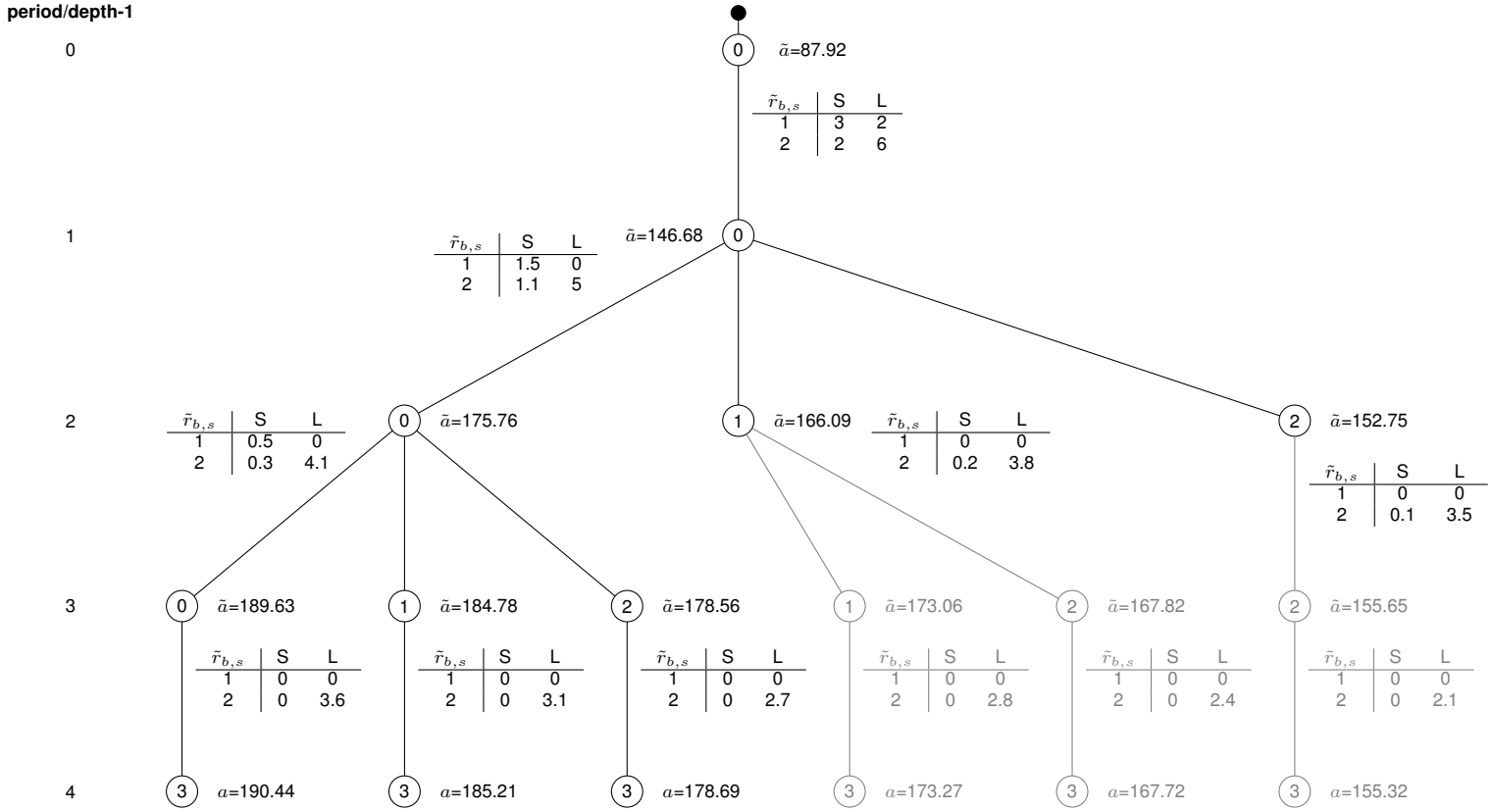

Figure 4.2: Example for the dynamic generation of mark-down strategies, Algorithm 3, POP-DYN

Instance	scenario low			scenario normal			scenario high		
	nd	d		nd	d		nd	d	
	time(s)	time(s)	%n	time(s)	time(s)	%n	time(s)	time(s)	%n
1	3.47	1.10	34.70	3.29	0.68	21.23	3.45	0.43	13.47
2	3.66	0.71	21.89	4.07	0.37	10.18	3.85	0.24	7.98
3	3.55	0.99	30.97	3.48	0.80	21.60	3.95	0.47	14.79
4	3.55	1.16	22.53	4.07	0.73	22.33	3.37	0.53	14.93
5	4.06	0.79	24.23	3.24	0.52	16.76	3.41	0.36	11.79
6	3.57	1.17	35.80	4.05	0.85	22.84	3.48	0.50	16.32
7	4.57	0.74	21.96	4.46	0.52	15.37	3.50	0.40	11.86
8	2.86	0.90	28.26	2.92	0.51	20.50	3.31	0.32	12.66
9	3.23	0.85	27.60	2.88	0.51	19.62	3.30	0.28	11.64
12	2.96	0.93	30.01	3.35	0.62	22.62	2.80	0.38	24.49
∅	3.55	0.93	27.80	3.58	0.61	19.31	3.44	0.39	13.99

Table 4.1: Dynamic generation of mark-down strategies – computational results

4.9 Computational results

In Table 4.1 we compare two implementations of Algorithm 3, one with the usage of dominance checks, Algorithm 5, and one without it. We took articles with known supply from our set $\mathcal{I}$, Appendix E, of real-world instances. We performed the tests for all three different scenarios, see also Chapter 3: good seller, bad seller and normal seller.

To compare the number of visited inner nodes we use the results of the corollaries 2, 3 and 4. We compare runtime in seconds "time" and the percentage of visited inner nodes of the enumeration tree.[1] With $k_{\max} = 13$ and $p_{\max} = 4$ the overall number of nodes in the enumeration tree according to Corollary 3 amounts to 1730. Without the leaves there remain $1730 - 364 = 1366 \mathrel{\widehat{=}} 100\%$ visited nodes in the complete enumeration tree for applying Algorithm 3 without dominance checks "nd". The percentages of this number for applying the procedure with dominance checks via Algorithm 5 "d" are given in the columns with label "%n".

We see that with the application of our dominance rule depending on the considered scenario we can reduce the mean number of visited inner nodes to a number between 12.83 and 27.55 percents of the number of inner nodes we would have to visit if we enumerated all trajectories. The computation times can be reduced by a factor between 3.38 and 8.29. For "higher" scenarios we get better results with the usage of dominance checks.

We used the same instances from the set $\mathcal{I}$ to compare Algorithm 3 with solving the mixed-integer programming formulation of Problem 5 directly via CPLEX. To avoid nonlinearity we set the mark-down costs – both, fixed and variable – to zero. The results for the "low" scenario can be found in Table 4.2.

While the solving process for the MIP averagely needs more than 8 hours our dynamic programming approach with dominance checks can yield the optimal solution averagely in less than one second. Compared to the results stated in Table 4.1 we also see that the number of cut off inner nodes is averagely more than 14 percentage points higher as against the case where mark-down costs are regarded.

[1] We exclude the leaves in the enumeration tree from this consideration because extensions to complete mark-down strategies are easy to compute by including the overall stock in the labels.

Instance	nd	d		MIP
	time(s)	time(s)	%nodes	time
1	3.56	0.48	14.79	30067.8
2	4.09	0.44	13.69	27423.3
3	4.08	0.44	15.45	28637.7
4	3.59	0.49	15.45	30303.6
5	3.49	0.53	14.79	28752.7
6	3.58	0.41	12.81	25800.5
7	4.56	0.18	4.90	36300.9
8	2.95	0.43	14.13	32898.0
9	2.99	0.36	13.47	30194.3
12	3.39	0.36	14.35	32356.2
∅	3.63	0.41	13.38	30273.5

Table 4.2: POP – dynamic generation versus MIP

4.10 Conclusion of the chapter

Every week our industrial partner for more than 4000 products has to decide on mark-downs. Hence, a fast approach for solving the Price Optimization Problem is necessary. In praxis there remain only two days to decide for an eventual mark-down. Our results show that therefore applying state-of-the-art solvers on the mixed-integer programming formulation of price optimization are not an alternative.

A standard approach for pricing problems is dynamic programming. We applied this idea on the Price Optimization Problem how it takes part in DISPO and extended the approach by dominance rules to a label setting algorithm. Now we can solve the Price Optimization Problem for one article averagely in less than one second. Our label setting algorithm with dominance checks against an enumeration of all mark-down strategies would reduce the runtime for deciding mark-downs for weekly 4000 articles from four hours to one hour. In this form our algorithm POP-DYN with dominance checks could be applied as a standard approach for deciding on mark-downs at our industrial partner.

Chapter 5

Stochastic Optimization

In this chapter we outline some basics of stochastic programming. We are mainly guided by [BL97].

In contrast to deterministic programs stochastic programs can contain random data. Thus, stochastic programming extends the field of mathematical programming by programming under uncertainty. Uncertain input data are reproduced by random variables with known distribution.

Stochastic programs are applied when not all of the input data is known at the time of decision making. The aim is, e.g., to optimize the expected costs over all possible scenarios.

Economical problems often have to deal with uncertainties. Demand for products, resources, etc. are not always known a priori. By treating this uncertainties in a stochastic program a more realistic problem formulation is anticipated which shall lead to better decisions at the end.

In Section 5.1 we will introduce so-called two-stage stochastic programs. Two-stage stochastic programs consist of a so-called first stage decision, a decision which has to be made before the scenario in effect is known. The second stage – or recourse decision – responds to the realization of the scenario. If the random events follow a discrete distribution with a finite number of scenarios the two-stage stochastic program can be formulated as a so-called deterministic equivalent. In Section 5.2 we sketch out approaches from literature to solve the deterministic equivalent. Our focus in this chapter is on dual bounds for general two-stage stochastic programs, Section 5.3 – later on we will apply them to the Integrated Size and Price Optimization Problem in the context of a customized Branch&Bound approach.

In Section 5.4 we sketch out the idea of multi-stage stochastic programs. Again a first-stage decision has to be made before anything about future behavior is known. But in contrast to two-stage programs – in which we only deal with one recourse decision – multi-stage programs involve sequences of recourse decisions over time depending on the realizations of the particular outcomes.

5.1 Two-stage stochastic programs

We start with an example of a popular stochastic program, the newsvendor problem, as it is for example stated in [BL97].

Example 6 (newsvendor problem)**.** In the morning a newsvendor buys x newspapers at a price c per paper from a publisher to sell them at the price of q on the street. The number x of bought newspapers is bounded above by u. The newsvendor sells as many papers as possible for the sales price q. At the end of the day he can return the remaining newspapers to the publisher at a price of r with $r < c$. The demand per day is varying and described by a random variable ξ.

The newsvendor problem can be formulated as a two-stage stochastic linear program. The *first stage* is the decision on how many newspapers the newsvendor should buy from the publisher. As *second stage* or *recourse* the newsvendor can compensate a wrong first-stage decision by returning overbought newspapers to the publisher.

We get to the general formulation of a two-stage stochastic program.

Definition 8 (two-stage stochastic program with recourse)**.**

$$\min_x c^T x + \mathbb{E}_\xi \mathcal{Q}(x, \xi) \tag{5.1}$$

$$\text{subject to } Ax = b, \tag{5.2}$$

$$x \geq 0. \tag{5.3}$$

It is

$$Q(x, \xi) = \min\{q_\xi^T y | W_\xi y = h_\xi - T_\xi x, x, y \geq 0\}. \tag{5.4}$$

The function $Q(x, \xi)$ is also called *recourse function*. The recourse is called *fixed* if the so-called *recourse matrix* W_ξ does not depend on any uncertainties, then it is $W_\xi = W$. The recourse is called *complete* if there is a valid second-stage decision for every first-stage decision and *relative complete* if for every valid first-stage decision for every scenario a valid second-stage decision exists. The matrix T is also denoted as *technology matrix*.

In the case that the random events follow a discrete distribution with a finite number of scenarios it is possible to reduce the two-stage stochastic program to a deterministic program. Then the expected value $\mathbb{E}_\xi Q(x, \xi)$ can be computed explicitely: For every variable y of the second stage one introduces a random variable for every particular scenario and obtains an equivalent linear program – the so-called *deterministic equivalent*.

Definition 9 (two-stage stochastic problem in its extensive form – deterministic equivalent)**.**

$$\min_x c^T x + \sum_{\xi \in \Xi} p_\xi q_\xi^T y_\xi \tag{5.5}$$

$$\text{subject to } Ax = b, \tag{5.6}$$

$$T_\xi x + W_\xi y_\xi = h_\xi, \quad \forall \xi \in \Xi, \tag{5.7}$$

$$x \geq 0, y \geq 0. \tag{5.8}$$

The set Ξ contains all scenarios which follow from the discrete distribution. The probability for the occurrence of scenario ξ is given by p_ξ.

The number of the second stage variables in terms of the classical formulation from Definition 8 in this models multiplies with the number of scenarios. This may lead to a large deterministic program. It may be that this deterministic equivalent can not be handled with standard approaches from linear programming.

5.2 Solving stochastic programs

In this section we sketch out common solving methods for two-stage stochastic programs. One of the most prominent method for linear programs (without integer variables) is the so-called L-shaped method. We outline the basic idea in Subsection 5.2.1. For general mixed-integer programs an important property the L-shaped method is based on is violated: The convexity of the recourse function. In literature applications of general approaches from mixed-integer programming like Branch&Bound are often proposed to handle two-stage mixed-integer linear programs. We state some references in Subsection 5.2.2.

5.2.1 The L-shaped method for two-stage linear stochastic programs

The L-shaped method for two-stage linear stochastic programs – in general also Benders' decomposition [Ben62] – exploits the special structure of the deterministic equivalent.[1]

The Bender's decomposition method is closely linked to the Dantzig-Wolfe decomposition [DW60]: It equals the Dantzig-Wolfe decomposition on the dual linear program: While in the case of a Dantzig-Wolfe decomposition in a column-generation algorithm [LD11] sequentially "promising" variables are added, in the L-shaped method step-by-step cuts are added.

For this purpose the L-shaped method exploits the property that the recourse function $Q(x, \xi)$ is piecewise linear and convex in x for a fixed ξ. Positive linear combinations of convex functions are convex and so the expected value $\mathcal{Q}(x) := \mathbb{E}_\xi(Q(x, \xi))$ is as well. The main idea of the L-shaped method is to approximate the term $\mathcal{Q}(x)$ in the objective function by piecewise linear functions.

Two types of constraints are sequentially added: On the on hand *optimality cuts* which are linear approximations of $\mathcal{Q}(x)$ and on the other hand *feasibility cuts* which restrict the set $\{x|Ax = b, x \geq 0\}$.

For details and the exact algorithm we refer the reader to [BL97], [KW94] or [Pré95].

5.2.2 Solving two-stage mixed-integer stochastic programs

In the case of (mixed-)integer stochastic programs the convexity of the function $\mathcal{Q}(x)$ – which is exploited by the L-shaped method – is no longer guaranteed.

In literature some extensions of the Benders' decomposition which deal with integrality of variables can be found. There are approaches for special problem structures. Wollmer [Wol80] extended Benders' decomposition for stochastic problems with binary first and continuous second stage variables. This algorithm was extended by Laporte and Louveaux [LL93] for binary first and binary or continuous recourse variables.

[1] The name L-shaped method stems from the block structure of the deterministic equivalent.

Carøe and Tind again extended this approach for continuous first stage and integer second stage variables. All these approaches and the historical progress are outlined by Sherali and Zhu [SZ09].

In recent times the L-shaped method is adapted to various special integer problems like dial-a-ride problems with stochastic customers delays [HCL11], stochastic Steiner tree problems [BCJ+10] or the traveling salesman problem with stochastic travel times [TOH04].

There are many references on solving mixed-integer stochastic programs with common approaches from mixed-integer programming. Schultz [Sch03] reviews existing approaches based on Branch&Bound, Lagrangian relaxation or cutting plane methods. Sherali and Zhu [SZ06] developed a Branch&Bound approach in which Benders' decomposition is used for dual bounding. There exists also literature about custom-made approaches for special problems, for example Branch&Bound algorithms for stochastic general assignment [ASvdVF06] or high speed telecommunication network design [ALMP05] problems.

5.3 Common bounds for two-stage stochastic programs

As outlined in the last section Branch&Bound is an frequently applied approach to tackle mixed-integer stochastic programs. The efficiency of this approach depends in a great part on the goodness of dual and primal bounds. Besides the common primal (valid solutions) and dual bounds (LP relaxations) for mixed-integer problems, in literature bounds especially for stochastic programs are proposed. At this point we will give an overview. More details can be found for example in [BL97].

5.3.1 Dual bounds

A well-known dual bound for two-stage stochastic programs is the so-called wait-and-see solution. Similar bounds can be derived by solving the so-called pairs subproblem which can be extended to the group subproblem.

The wait-and-see solution

To illustrate the idea behind this bound let us assume that already at the time when the first stage decision has to be made, we know which scenario will occur. Then the logical consequence would be to determine the first stage decisions optimal in terms of the scenario in effect.[2]

For each scenario we can compute an optimal first stage decision a priori. Because no compromises in terms of the other scenarios have to be made the particular first stage decision yields lower costs for the corresponding scenario than the optimal first stage decision of the original problem.

To formalize this idea we consider the formulation of a general two-stage stochastic program from Definition 8 for only the particular scenario ξ, i.e. we set the probability of ξ to value one and the probability of all other scenarios to value zero.

[2]The name wait-and-see solution can be derived as follows: One *waits* until he will *see* which scenario occurred and only then the first stage decision is made.

Definition 10 (two-stage linear stochastic program associated with one scenario)**.**

$$\min_x c^T x + Q(x, \xi) \tag{5.9}$$

$$\text{subject to } Ax = b, \tag{5.10}$$

$$x \geq 0. \tag{5.11}$$

with $Q(x, \xi)$ as in Definition 8. The optimal solution is denoted by $\bar{x}$. It is $z(\bar{x}, \xi)$ the related optimal objective value.

The wait-and-see solution WS is defined as

$$WS := \mathbb{E}_\xi z(\bar{x}, \xi). \tag{5.12}$$

Birge and Louveaux showed that the wait-and-see solution of a general two-stage stochastic linear program is a dual bound for the original problem.

Theorem 4 (wait-and-see solution as dual bound [BL97])**.** *With RP being the optimal objective function value of the two-stage stochastic program from Definition 8 it is*

$$WS \leq RP. \tag{5.13}$$

Proof. [BL97] With $z(x^*(\Xi), \xi)$ we denote the objective value that results from applying the optimal solution x^* of the two-stage stochastic program from Definition 8 on the program from Definition 10. By definition it is $z(\bar{x}, \xi) \leq z(x^*(\Xi), \xi)$. Applying the expected value over all scenarios $\xi \in \Xi$ on both sides yields the claim. □

To compute the wait-and-see solution for a deterministic equivalent one has to consider $|\Xi|$ linear programs where the number of second stage variables of a particular problem reduces by factor $|\Xi|$ against the original problem from Definition 8. Therefore for large problems the wait-and-see solution might be easier to compute than the optimal solution of the original problem.

There is a descriptive interpretation for the difference between the value of the wait-and-see solution and the optimal objective value of the original problem also called *expected value of perfect information* $EVPI$. The $EVPI$

$$EVPI = RP - WS, \tag{5.14}$$

measures how useful the knowledge about the scenario in effect is. So vividly it can be seen as the maximum price one would pay to a psychic to obtain reliable information about the occurring scenario.

Pairs subproblems

Bounds similar to the wait-and-see solution can be derived by solving so-called *pairs subproblems*. For a pairs subproblem a *reference scenario* ξ_r is chosen. As above we denote the scenario probabilites by p_{ξ_i}. One could also choose a reference scenario which is not contained in the set Ξ. In this case the probability p_{ξ_r} would take value zero.

Definition 11 (pairs subproblem)**.** The pairs subproblem for the reference scenario ξ_r and the scenario $\xi_k \in \Xi$ is defined as

$$z_P(x, \xi_r, \xi_k) := \min_x c^T x + p_{\xi_r} q_{\xi_r}^T y_{\xi_r} + (1 - p_{\xi_r}) q_{\xi_k}^T y_{\xi_k} \tag{5.15}$$

$$\text{subject to } Ax = b, \tag{5.16}$$

$$W_{\xi_r} y_{\xi_r} = h_{\xi_r} - T_{\xi_r} x, \tag{5.17}$$

$$W_{\xi_k} y_{\xi_k} = h_{\xi_k} - T_{\xi_k} x, \tag{5.18}$$

$$x, y \geq 0. \tag{5.19}$$

The pairs subproblem can be seen as a stochastic program with only two possible realizations ξ_r with probability p_r and ξ_k with probability p_k.

The *sum of pairs expected value* $SPEV$ is given by

$$SPEV = \frac{1}{1 - p_{\xi_r}} \sum_{\xi_k \in \Xi, \xi_k \neq \xi_r} z_P(x, \xi_r, \xi_k). \tag{5.20}$$

If $\xi_r \notin \Xi$, then the sum of pairs expected values is equivalent to the wait-and-see solution.

Additionally it is

$$WS \leq SPEV \leq RP. \tag{5.21}$$

Birge and Louveaux [BL97] proved these inequalities similarly to Theorem 4. To obtain the first inequality they compared the optimal objective value of the problem from Definition 10 for the reference scenario ξ_r (which is also feasible for the pairs subproblem) with the optimal objective value of the pairs subproblem.

For the second inequality they compared the optimal objective value of the original problem with the optimal objective value of the pairs subproblem. They exploited that the optimal first and particular second stage decisions for the scenarios ξ_r and ξ_k are also feasible for the pairs subproblem.

Group subproblems

An extension of the pairs subproblems are the so-called *group subproblems*. Here a reference scenario together with a subset of scenarios is considered. We found two different formulations of group subproblems. One from Birge and Louveaux [Bir82] and a recent formulation by Sandikçi et al. [SKS12]. Sandikçi et al. showed by counterexamples that the formulation of [Bir82] in general does not yield correct dual bounds. Therefore we focus on the group subproblem as it is stated in [SKS12].

Again we consider a reference scenario ξ_r with probability p_{ξ_r} which not necessarily has to be from the set Ξ. It is $S := \{1, 2, \ldots, K\}$ the index set of scenarios excluding ξ_r and $\mathcal{P}(S)$ the power set of S without the empty set. For $k = 1, 2, \ldots, K$ the set of all subsets of $\mathcal{P}(S)$ with k elements is given by $\mathcal{P}_k(S) := \{\Gamma \in \mathcal{P}(S) : |\Gamma| = k\}$. It is $\rho(\Gamma) := \sum_{j \in \Gamma} p_{\xi_j}$ the sum of probabilities of all scenarios included in Γ.

Definition 12 (group subproblem $GR(\Gamma)$)**.** The group subproblem $GR(\Gamma)$ for the reference scenario ξ_r and the subset of scenarios $\Gamma \in \Xi$ is given by

$$z^*(\Gamma) = \min_x c^T x + p_{\xi_r} q_{\xi_r}^T y_{\xi_r} + (1 - p_{\xi_r}) \left(\sum_{i \in \Gamma} \frac{p_{\xi_i}}{\rho(\Gamma)} q_{\xi_i}^T y_{\xi_i} \right) \tag{5.22}$$

$$\text{subject to } Ax = b, \tag{5.23}$$

$$W_{\xi_r} y_{\xi_r} = h_{\xi_r} - T_{\xi_r} x, \tag{5.24}$$

$$W_{\xi_i} y_{\xi_i} = h_{\xi_i} - T_{\xi_i} x, \quad \forall i \in \Gamma, \tag{5.25}$$

$$x, y \geq 0. \tag{5.26}$$

To obtain the dual bound named *expected value of the group subproblem EGSO(k)* all group subproblems for subsets of k scenarios have to be solved and the expected value of the optimal objective values has to be taken where the probability assigned to a group subproblem $GR(\Gamma)$ with $|\Gamma| = k$ is given by $\frac{\rho(\Gamma)}{\sum_{\Gamma \in \mathcal{P}_k(S)} \rho(\Gamma)}$.

The authors exploited the fact that a scenario index i is contained in exactly $\binom{K-1}{k-1}$ elements of $\mathcal{P}_k(S)$.

Then for $1 \leq k \leq K$ it is

$$\sum_{\Gamma \in \mathcal{P}_k(S)} \rho(\Gamma) = \binom{K-1}{k-1} \rho(S) = \binom{K-1}{k-1} \cdot \sum_{i \in S} p_{\xi_i} = \binom{K-1}{k-1} (1 - p_{\xi_r}) \tag{5.27}$$

$EGSO(k)$ is then defined as

$$EGSO(k) := \frac{1}{\binom{K-1}{k-1} (1 - p_{\xi_r})} \sum_{\Gamma \in \mathcal{P}_k(S)} (\rho(\Gamma) \cdot z^*(\Gamma)). \tag{5.28}$$

The authors prove that there is a hierarchy of bounds depending on the size of the considered subsets of scenarios, namely

$$WS \leq EGSO(1) \leq EGSO(2) \leq \ldots \leq EGSO(K-1) \leq EGSO(K) = RP. \tag{5.29}$$

The proof is divided into three steps. It is shown that

1. $WS \leq EGSO(1)$,
2. $EGSO(k) \leq EGSO(k+1)$,
3. $EGSO(K) = RP$.

The first claim follows from the observation that $EGSO(1)$ is equivalent to $SPEV$. As already mentioned the inequality $SPEV \leq WS$ has been proved by Birge and Louveaux [BL97].

The third claim follows easily by the definition of the group subproblem: $EGSO(K)$ equals the original problem from Definition 8.

For the proof of the second inequality the authors exploit some Lemmas in terms of properties of subsets of scenario index sets and optimal solutions of the group subproblem. For further reading and a formal proof we refer the reader to [SKS12].

5.3.2 Primal bounds

We will present some primal bounds from literature particular for stochastic programs. All outlined approaches have one similarity: The random variables are replaced by their expected values.

EEV

The *expected value problem EV* is the problem that arises by replacing all random variables in the original problem from Definition 8 by their expected value $\bar{\xi} := \mathbb{E}(\xi)$. That means we consider a problem with the same constraints as the original problem regarding only one scenario with a probability of value one.

Formally the objective of the *expected value problem* is given by

$$\min_x c^T x + Q(x, \bar{\xi}). \tag{5.30}$$

The expected value problem yields a first stage decision. To obtain corresponding second stage decisions for every particular scenario a second optimization is performed. This is done by fixing the according to the objective function (5.30) optimal first stage decisions $\bar{x}(\bar{\xi})$ in the original problem from Definition 8.

We denote the corresponding optimal objective value with $z(\bar{x}(\bar{\xi}), \xi)$.

The *expected result of using the EV solution EEV* is given by

$$EEV = \mathbb{E}_\xi(z(\bar{x}(\bar{\xi}), \xi)). \tag{5.31}$$

It is

$$RP \leq EEV. \tag{5.32}$$

This inequality is easily to comprehend if we realize that $\bar{x}(\bar{\xi})$ is only one specific solution of the original problem. It is EEV the related objective value for this solution, while RP is the optimal objective value.

We can interpret the difference, also called the *value of stochastic solution* VSS, between the EEV and the optimal objective function value RP of the original problem, i.e.

$$VSS = EEV - RP, \tag{5.33}$$

as the cost for not regarding stochastic influences.

Further bounds

Birge and Louveaux showed that for two-stage stochastic programs with only continuous variables the pairs subproblem in which all random variables are replaced by their expected value yields a primal bound named $EPEV$ with $EPEV \leq EEV$. For details see [Bir82] or [BL97].

Similarly Sandikçi et al. [SKS12] derived an upper bound based on group subproblems for two-stage stochastic mixed-integer programs.

5.4 Multi-stage stochastic programs

Until now we have only considered two-stage stochastic programs where a recourse decision is only made one time. But sometimes it could be useful that problems involve sequences of decisions over time that are dependent on realizations of outcomes that are a priori unknown. In this case one would formulate a so-called *multi-stage* stochastic program. For illustration we consider a lightly modified version of the newsvendor problem.

Example 7 (newsvendor problem with return)**.** At the beginning of the day a newsvendor buys x papers at a price c per paper from the publisher to sell them on the street. The amount x of bought papers is bounded above by u. The newsvendor sells as many papers as possible for price q. The sales until the middle of the day follow a stochastic demand ξ_1. In the middle of the day he either can return the remaining newspapers to a price r_1 with $r_1 < c_1$ to the publisher or decide to try to sell the remaining items at the remainder of the day. The demand until evening is given by a random distribution ξ_2. At the end of the day the newsvendor can return non-sold items for price r_2 with $r_2 < r_1$ to the publisher.

In this example there are three stages:

1. In the morning of the first day, the vendor decides on how many items to buy from the publisher.
2. In the middle of the day, he decides on how many items to return the publisher.
3. At the end of the day, remaining items can be returned to the publisher.

In this case we have a multi-stage stochastic optimization problem, exactly a three-stage one. Now there are two recourse stages and both depend on the previous scenarios in effect.

Multi-stage stochastic optimization problems are characterized by the occurrence of the random events in a chronological process where new decisions have to be made meanwhile.

This model would extend to n stages if the newsvendor could return the remaining items $n-1$ times a day to the publisher for price $r_i, i = 1, \ldots, n$ with $r_n < r_{n-1} < \ldots < r_1 < r_0 < c_0$.

However, our focus lies on two-stage models and for further information about multi-stage models we refer the reader to [BL97], [Pré95] or [KW94].

Chapter 6

The Integrated Size and Price Optimization Problem (ISPO)

In this chapter we present our model ISPO for integrated size and price optimization. The model is an advancement of the model SLDP, presented in Chapter 2. Because demand is a priori unknown and depends on the scenario in effect, ISPO is formulated as a stochastic program. More precisely, a two-stage stochastic program with fixed recourse where the so-called *first-stage decision* is the supply in terms in lots and the *recourse* the price optimization where oversupply is compensated by marking down prices. The target is to find a supply policy that maximizes the expected profit over all scenarios – "bad seller", "normal seller" and "good seller". We will specify the problem in Section 6.1 and outline ISPO in its extensive form in Section 6.2. In Section 6.3 we show that ISPO is NP-hard by reducing to it the SLDP. In Chapter 4 we presented a dynamic programming approach for solving the Price Optimization Problem where the supply is fixed. One could also think about applying dynamic programming for solving ISPO for all possible supply strategies. In Section 6.4 we will see that because of the special situation at our industrial partner – the supply in terms of lots – the state space is too large to apply dynamic programming. Moreover state-of-the-art MIP solvers cannot deal with the problem size of ISPO.

6.1 Problem specification

An instance of ISPO among others consists of a set B of *branches*, a set L of *lot-types* and a set $M = \{1, \dots, m^{\max}\}$ of *multiplicities* for the given lot-types. Also a *set of sizes* S is specified. With l_s we denote the number of items of Size s in Lot-type l.

The set of lot-types is given by four parameters. The *minimum number of items per size* $v^{\min}$ with $v^{\min} \geq 1$, the *maximum number of items per size* $v^{\max}$, the *minimum number of items per lot-type* $vl^{\min}$ and the *maximum number of items per lot-type* $vl^{\max}$.

An upper bound for the *maximum number of supplied different lot-types* κ is given. *Lot opening costs* δ_i for using an additional i-th lot-type $i = 1, \dots, \kappa$ are also specified. Moreover *pick costs* pcost arise for each supplied lot-type. An *acquisition price* ap has to be paid for each supplied item. The *lower* and *upper bounds for overall supply* are given by $\underline{I}$ and $\overline{I}$.

In terms of the sales success of the considered article a *set of scenarios* E with *scenario probabilites* $\mathrm{Prob}(e), \forall e \in E$ is given. A set of *sales periods* $K = \{0, \ldots, k_{\max}\}$ is specified. We call $0, \ldots, k_{\max} - 1$ the *real sales periods* where $k_{\max}$ is the *sellout period*. For a duration of k_{obs} *observation periods* the article will not be marked down.

We are given a set of *price indices* $P = \{0, \ldots, p_{\max}\}$ where the related *price steps* $\pi_p, p \in P$ are in descending order. The first price step π_0 is the starting price, the last price step $\pi_{p_{\max}}$ is the *salvage value*.

Moreover a factor ρ for weekly discounting is given. For every mark-down uniquely fixed mark-down cost μ_f and mark-down cost μ_v per marked down item arise. In the sellout period $k_{\max}$, where all remaining items are sold, we assume that every left over item has to be marked down $q_{k_{\max}}$ times. Here only variable mark-down costs μ_v per item accrue. For every scenario e, period k, branch b, size s and price index p the *dependent demand* is given by $d^e_{k,b,s,p}$.

Our goal is to maximize the expected revenue for supplying each branch with a lot-type in a multiplicity by taking into account that mark-downs may occur during the selling time.

6.2 ISPO as a two-stage stochastic mixed-integer program (SMIP) in its extensive form

Now we present one of the main aspects of this work – our formulation of the Integrated Size and Price Optimization Problem ISPO.

We start with a look at the objective function coefficients.

For the objective function we introduce handling costs $c_{b,\ell,m}$. They are computed as described in the following definition.

Definition 13 (handling costs $c_{b,\ell,m}$)**.** For a given acquisition price ap for one item and given pick cost pcost for one lot-type the handling costs for supplying Branch b with Lot-type ℓ in Multiplicity m are given by

$$c_{b,\ell,m} := m \cdot \left(\sum_{s \in S} l_s \mathrm{ap} + \mathrm{pcost} \right). \tag{6.1}$$

For the first stage – the size optimization stage (SOP) – we use binary assignment variables $x_{b,\ell,m}$ to encode the independent assignment of Branch b to Lot-type ℓ in Multiplicity m. In the second stage – the price optimization stage (POP) – we introduce binary assignment variables for the independent second stage assignment decision $u^e_{k,p}$of Price index p to Period k in Scenario e. In order to account for the profit and the cost, we need additional dependent variables. We list the complete model before we comment on the details.

Problem 6 (ISPO)**.**

$$\max - \sum_{b \in B} \sum_{\ell \in L} \sum_{m \in M} x_{b,\ell,m} \cdot c_{b,\ell,m} - \sum_{i=1}^{\kappa} \delta_i \cdot z_i \tag{6.2}$$

$$+ \sum_{e \in E} \mathrm{Prob}(e) \sum_{k \in K} \exp(-\rho k) \Big(\sum_{b \in B} \sum_{s \in S} r^e_{k,b,s} - \mu_k \beta^e_k \Big) \tag{6.3}$$

Size Optimization Stage (SOP):

$$\sum_{\ell \in L} \sum_{m \in M} x_{b,\ell,m} = 1 \quad \forall b \in B, \tag{6.4}$$

$$\sum_{m \in M} x_{b,\ell,m} \leq y_\ell \quad \forall b \in B, \ell \in L, \tag{6.5}$$

$$\sum_{\ell \in L} y_\ell \leq \sum_{i=1}^{\kappa} z_i, \tag{6.6}$$

$$z_i \leq z_{i-1} \quad i = 1, \ldots, \kappa, \tag{6.7}$$

$$I_{b,s} = \sum_{\ell \in L} \sum_{m \in M} m \cdot \ell_s \cdot x_{b,\ell,m} \quad \forall b \in B, s \in S, \tag{6.8}$$

$$I = \sum_{b \in B} \sum_{s \in S} I_{b,s}, \tag{6.9}$$

$$I \in [\underline{I}, \overline{I}], \tag{6.10}$$

$$x_{b,\ell,m} \in \{0,1\} \quad \forall b \in B, \ell \in L, m \in M, \tag{6.11}$$

$$y_\ell \in \{0,1\} \quad \forall \ell \in L, \tag{6.12}$$

$$z_i \in \{0,1\} \quad \forall i = 1, \ldots, \kappa, \tag{6.13}$$

Coupling via initial inventory:

$$I_{b,s} - v^e_{0,b,s} = 0, \quad \forall b \in B, s \in S, e \in E, \tag{6.14}$$

Price Optimization Stage (POP):

$$\sum_{p \in P} u^e_{k,p} = 1 \quad \forall k \in K, e \in E, \tag{6.15}$$

$$u^e_{k,0} = 1 \quad \forall k \in K : k < k_{obs}, e \in E, \tag{6.16}$$

$$u^e_{k_{\max},p_{\max}} = 1 \quad \forall e \in E, \tag{6.17}$$

$$u^e_{k-1,p_1} + u^e_{k,p_2} \leq 1 \quad \forall k \in K, e \in E, p_1, p_2 \in P : p_2 < p_1, \tag{6.18}$$

$$\beta^e_k \geq u^e_{k-1,p_1} + u^e_{k,p_2} - 1 \quad \forall k \in K, e \in E, p_1, p_2 \in P : p_2 \neq p_1, \tag{6.19}$$

$$v^e_{k-1,b,s} - v^e_{k,b,s} = \sum_{p \in P} w^e_{k-1,b,s,p} \quad \forall k \in K, b \in B, s \in S, e \in E, \tag{6.20}$$

$$\sum_{p \in P} w^e_{k,b,s,p} \leq v^e_{k,b,s} \quad \forall k \in K, b \in B, s \in S, e \in E, \tag{6.21}$$

$$w^e_{k,b,s,p} \leq u^e_{k,p} \cdot d^e_{k,p,b,s} \quad \forall k \in K, b \in B, s \in S, p \in P, e \in E, \tag{6.22}$$

$$r^e_{k,b,s} = \sum_{p \in P} \pi_p \cdot w^e_{k,b,s,p} \quad \forall k \in K, b \in B, s \in S, e \in E, \tag{6.23}$$

$$u^e_{k,p} \in \{0,1\} \quad \forall k \in K, p \in P, e \in E, \tag{6.24}$$

$$\beta^e_k \in \{0,1\} \quad \forall k \in K, e \in E, \tag{6.25}$$

$$w^e_{k,b,s,p} \geq 0 \quad \forall k \in K, b \in B, s \in S, p \in P, e \in E, \tag{6.26}$$

$$v^e_{k,b,s} \geq 0 \quad \forall k \in K, b \in B, s \in S, e \in E, \tag{6.27}$$

$$r^e_{k,b,s} \geq 0 \quad \forall k \in K, b \in B, s \in S, e \in E, \tag{6.28}$$

$$\mu_k = \mu_f + \mu_v \sum_{b\in B}\sum_{s\in S} v^e_{k,b,s}, \quad \forall k \in K \setminus \{k_{\max}\}, e \in E, \tag{6.29}$$

$$\mu_{k_{\max}} = q_{k_{\max}}\mu_v \sum_{b\in B}\sum_{s\in S} v^e_{k_{\max},b,s}, \quad \forall e \in E. \tag{6.30}$$

We start our explanations with the SOP stage: The constraints are the same as in the formulation of the SLDP (Problem 2). For the sake of completeness we will go into them anyway. The binary variables $x_{b,\ell,m}$ indicate if Lot-type ℓ is delivered to Branch b in Multiplicity m. If this is the case they take value one, and zero otherwise. We force an assignment of a lot-type and a multiplicity to each branch by Equation (6.4). In order to account for the opening costs of the supplied lot-types, we introduce binary variables y_ℓ indicating whether or not lot-type ℓ is used at all and binary variables z_i that take value one if and only if at least i different lot-types are used for supply. Equation (6.5) guarantees that $y_\ell = 1$ whenever ℓ is assigned to at least one branch b. Inequality (6.6) implies that no more than κ lot-types are used. Inequality (6.7) enforces that $z_i = 1$ implies that the number of used lot-types is at least i. We use another dependent variable $I_{b,s}$ for the inventory in branch b and size s, and Equation (6.8) links this variable to the assignment decisions. The total inventory is then given by yet another dependent variable I, computed by Equation (6.9) and enforced to lie in between given bounds by Inequality (6.10). All independent variables have to be binary, see (6.11) through (6.13).

Next, let us have a look at the POP stage model that is linked via the start inventories $I_{b,s}$ to the SOP stage by Equation (6.14).

The binary variables $u_{k,p}$ indicate if Price index p is allocated to Period k. If this is the case, they take value one, and zero otherwise. Equation (6.15) enforces the assignment of exactly one price to each period for each particular scenario. For a given number of periods k_{obs} from the beginning of the sales process the starting price is enforced, Equation (6.16). In the last period – the sellout – the salvage value is fixed by Equation (6.17). We forbid increasing prices by Equation (6.18). A mark-down for Scenario e in Period k is indicated by the dependent binary variable β^e_k, which is forced to one by Inequality (6.19) if the price has changed compared to the previous period. The mark-down costs for the real sales periods are given by Equation (6.29), the mark-down costs for the sellout period by Equation (6.30).

The following restrictions model the dynamics of the sales process using dependent variables. The fractional variable $v^e_{k,b,s}$ specifies the stock level in Period k in Branch b for Size s in Scenario e. The fractional variable $w^e_{k,b,s,p}$ measures the sales in Period k in Branch b and Size s for the price with index p in Scenario e. We capture the yield $r^e_{k,b,s}$ in Period k for Size s in Branch b in Scenario e by Equation (6.23). Equation (6.20) describes the change of stock levels from one period to another. Inequality (6.21) models that sales may not exceed stock. In Inequality (6.22) we require that, only if the price with related price index p is chosen, there can be sales at the price index p of at most the demand at the price index p. Because the objective favors larger sales, the sales variables at a price in an optimal solution will attain exactly the minimum of stock and demand at that price.

In this POP stage, only the independent price assignment variables need to be binary (6.24). The dependent variables capturing the dynamics of mean stocks, sales, and yields are required to be nonnegative in (6.26) through (6.28).

The objective function subtracts the costs for the handling of m lots of Type ℓ in Branch b and the lot-type opening costs for using the first, second, ..., i-th new lot-type (6.2) from the expected discounted yields minus the expected discounted costs for

mark-downs, (6.3).

6.3 Complexity of ISPO

The complexity of ISPO can be derived by reducing the SLDP on it.

Corollary 5 (Complexity of ISPO)**.** *ISPO is NP-hard.*

Proof. If we set all prices π_p for all $p = 0, \dots, p_{\max}$ and the fixed and variable mark-down costs in ISPO to value zero then the term

$$\sum_{e \in E} \text{Prob}(e) \sum_{k \in K} \exp(-\rho k) \left(\sum_{b \in B} \sum_{s \in S} r^e_{k,b,s} - \mu_k \beta^e_k \right)$$

in the objective will take value zero. (This would be the same as completely ignoring the price optimization stage in ISPO). The constraints of the size optimization stage still have to be fulfilled. Because these constraints are the same as for the SLDP we can solve each instance of the SLDP by setting $c_{b,\ell,m} = \text{dist}^{\text{SLDP}}_{b,\ell,m} + m \cdot \text{pcost}$. In Corollary 1 we mentioned that the SLDP is NP-hard and therefore it follows that ISPO is too. □

6.4 Solving ISPO with standard approaches

In Chapter 4 we presented a dynamic programming approach for the Price Optimization Problem. With a given supply per branch and size as initial state and applying our dominance rules we can solve the problem for all tested instances in less than four seconds to optimality. One could also think about applying dynamic programming to ISPO. In ISPO the initial supply for each branch and size is unknown from the beginning. The optimal supply is the supply that yields – together with the corresponding locally optimal mark-down strategies – the highest expected revenue. For each particular supply strategy theoretically a dynamic programming approach analogous to Chapter 4 could be applied. Finally we would choose the supply which maximizes the expected revenue in terms of the locally optimal price trajectories. But the supply has to be in lot-types and the single branches are connected by the overall supply I and the maximum number κ of used lot-types, Constraints (6.6) and (6.10). Moreover mark-downs are applied simultaneously over all branches and sizes. Therefore the state space would be huge. To get an idea how large the state space is, let us consider a small example. As in practice we are given $|B| = 1300$ branches, but consider only lot-types $(1, 2, 1), (2, 1, 1)$ and $(1, 1, 2)$, set $M = \{1\}$ and restrict the overall supply by $\underline{I} = 5200$ and $\overline{I} = 5200$. We set $\kappa = 3$.[1] Because every of our 1300 branches can be supplied by each lot-type this yields $3^{1300} > 10^{620}$ different supply policies. In fact, for real instances with about 729 different lot-types, at least 3 multiplicites and larger ranges in terms of the overall supply our state space is much bigger. So there is no possibility to solve ISPO just by dynamic programming to optimality. However dynamic programming is part of our heuristic solver for ISPO which we will present in Chapter 4. Here we restrict ourselves on small subsets of "most promising" lot-types.

[1] In fact these are no real restrictions. Consider the cardinality of the lot-types, the fact that the sole multiplicity takes value one and the number of branches.

Without the fixings (6.16), (6.17) and obviously redundant constraints (6.8), (6.9), (6.23) the number of constraints of ISPO amounts to

$$|E|(|B||S| + |K|(1 + 1.5|P|(|P| - 1) + |B||S|(1 + |P|))) + |B|(1 + |L|) + 2 + \kappa.$$

Without the obviously redundant variables (I,$I_{b,s}$,$r^e_{k,b,s}$) there are

$$|L|(|B||M| + 1) + \kappa + |E||K|(1 + |B||S|)(|P| + 1)$$

variables.

If we considered practical relevant instances with 3 scenarios, 1300 branches, 6 sizes, 14 periods, 5 prices, 729 lot-types, 3 multiplicities and maximal $\kappa = 4$ allowed different lot-types this would lead to 1 990 308 constraints and 4 809 685 variables. For typical computers (8 GB of RAM) the size of these instances exceeds memory capacity and CPLEX fails already at the initialization.

An instance of this size was tested on a machine with 128 GB of RAM.[2] Even after four weeks CPLEX was not able to solve the root relaxation. For an instance with 12 periods, 1000 branches, 7 lot-types, 5 multiplicities, maximal $\kappa = 5$ allowed used lot-types and the rest as above CPLEX needed more than 4 weeks to get to a solution with an optimality gap of 11.33%. So we tried a very small instance with only 30 branches, 5 periods and 435 lot-types, 3 multiplicities and maximal $\kappa = 4$ allowed lot-types and the rest as above.[3] The optimal solution is found after about five hours but still, after more than four weeks, there is no proof for optimality.

To put this into perspective: Our Branch&Bound solver presented in Chapter 9 is able to solve such instances in less than 30 seconds. And, more importantly, enables us to tackle real-world instances.

[2] Quad-Core AMD Opteron(tm) Processor 2384 CPU with 2GHz and 128 GB of RAM

[3] It is the first instance of our test set $\mathcal{I}_6^{\text{test}}$, see Appendix E.

Chapter 7

Reducing ISPO to the SLDP

As we have seen in Chapter 6, state-of-the-art solvers fail for real instances of ISPO. In Chapter 4 we deduced the number of possible price trajectories for the POP$^{\hat{e}}$. With this relatively small number – beside dynamic programming – it is possible to enumerate all valid price trajectories a priori. We exploit this property and in the remainder of this thesis only consider price trajectories instead of single assignments "price index to period".

We will show that fixing a price trajectory to each scenario simplifies ISPO to an SLDP with changed objective coefficients. To obtain these, in Section 7.1 we compute for each number of supplied items per branch and size the expected revenue – we call it *single supply revenue*. Adding up corresponding single supply revenues leads to so-called *lot-type* revenues which are objective function coefficients in the resulting SLDP, Section 7.2.

We state the SLDP that results by fixing price trajectories to scenarios in ISPO in Section 7.3.

In Section 7.4 we will show how ISPO *theoretically* could be solved by enumerating SLDPs for all possible assignments "scenario to price trajectory".

7.1 Single supply revenues

Now we consider the case that in ISPO a price trajectory to a particular scenario is fixed. For the fixed price trajectory we are able to compute the expected revenue for the related scenario for every branch, size and integer supply – the *single supply revenue* in advance.

Before we describe the algorithm for the computation of the single supply revenues we introduce some notation.

Notation 1 (map scenario to price trajectory, set of maps)**.** For a scenario e and a price trajectory t we call $e \to t$ a *map* from scenario e to price trajectory t. A *map* $e \to t$ means that we fix Price trajectory $t = (t_0, \ldots, t_{k_{\max}})$ to Scenario e in ISPO. In the formulation of Problem 6 this would mean setting the corresponding $u^e_{k,p}$ with $t_k = p$ to 1, $\forall k \in K$ and the remaining $u^e_{k,p}$ to 0. With $W^{E'}$ we denote a *set of maps* for a subset $E' \subseteq E$ of all scenarios, where each scenario $e \in E'$ is mapped to exactly one valid price trajectory.

Because the sales among the branches and sizes are independent from each other we can determine the expected revenue for each branch and size separately.

Our computation of the revenue does not regard the fixed costs for mark-downs a priori. Because they only depend on the corresponding price trajectory and not on the sold items or the stock after a period we will treat them later on.

We define the single supply revenue as follows.

Definition 14 (single supply revenue)**.** We consider a map scenario to price trajectory $e \to t$. For Branch $b \in B$ and Size $s \in S$ the *single supply revenue* $\bar{a}^{e\to t}_{b,s,n}$ for a number n of supplied items is defined as

$$\begin{aligned}\bar{a}^{e\to t}_{b,s,n} :=& -n \cdot \mathrm{ap} \\ &+ \sum_{k=0}^{k_{\max}-1} \exp(-\rho k)\left(\pi_{t_k} \min\left\{\max\left\{n - \sum_{j=0}^{k-1} d^e_{j,b,s,t_j}, 0\right\}, d^e_{k,b,s,t_k}\right\}\right. \\ &\left. - \beta_k \mu_v \max\left\{n - \sum_{j=0}^{k-1} d^e_{j,b,s,t_j}, 0\right\}\right) \\ &+ \exp(-\rho k_{\max})\left((\pi_{k_{\max}} - q_{k_{\max}}\mu_v)\max\left\{n - \sum_{k=0}^{k_{\max}-1} d^e_{k,b,s,t_k}, 0\right\}\right)\end{aligned} \tag{7.1}$$

For supplying 0 items the single supply revenue $\bar{a}_{b,s,0}$ takes value 0. For each additional supplied item we have to pay the acquisition price and the markdown costs. The more items are supplied the higher the costs for markdowns and for acquisition are. Because of the non-increasing prices and the discounting the yield we earn for each more supplied item will never exceed the yield we obtained for the last item.

The single supply revenue $\bar{a}^{e\to t}_{b,s,n}$ is concave in n.

Theorem 5 (concavity of the single supply revenue)**.** $\bar{a}^{e\to t}_{b,s,n}$ *is concave for* $n \geq 0$.

Proof. We show that the function is concave for all real values $\tilde{n}$ with $\tilde{n} > 0$. Then we can transfer the concavity to n with $n > 0$ and n integer.

As a linear function $-\tilde{n} \cdot \mathrm{ap}$ is concave.

The term

$$\exp(-\rho k)\left(-\beta_k \mu_v \max\left\{\tilde{n} - \sum_{j=0}^{k-1} d^e_{j,b,s,t_j}, 0\right\}\right)$$

is concave: As a linear function $\tilde{n} - \sum_{j=0}^{k-1} d^e_{j,b,s,t_j}$ is convex. Likewise the constant function 0 is. Maxima of convex functions are also convex, such is

$$\max\left\{\tilde{n} - \sum_{j=0}^{k-1} d^e_{j,b,s,t_j}, 0\right\}.$$

It is $\exp(-\rho k)$, β_k and $\mu_v \geq 0$. Multiplying a convex function with positive scalars does not change the convexity. But negative convex functions are concave. This yields the claim.

For the same reasons

$$\exp\left(-\rho k_{\max}\right)\left(-q_{k_{\max}}\mu_v \max\left\{\tilde{n} - \sum_{k=0}^{k_{\max}-1} d^e_{k,b,s,t_k}, 0\right\}\right)$$

is a concave function in $\tilde{n}$.

We now show that

$$\sum_{k=0}^{k_{\max}-1} \exp\left(-\rho k\right)\pi_{t_k} \min\left\{\max\left\{\tilde{n} - \sum_{j=0}^{k-1} d^e_{j,b,s,t_j}, 0\right\}, d^e_{k,b,s,t_k}\right\}$$
$$+ \exp\left(-\rho k_{\max}\right)\pi_{k_{\max}} \max\left\{\tilde{n} - \sum_{k=0}^{k_{\max}-1} d^e_{k,b,s,t_k}, 0\right\}$$

is concave. Because positive linear combinations of concave functions are also concave the claim of the theorem follows.

To simplify the term we denote $\exp(-\rho k)\,\pi_{t_k}$ by $\tilde{\pi}_k,\ \forall k \in K$. Because we consider only one branch, size and scenario we can simplify d^e_{k,b,s,t_k} by $\tilde{d}_k$. Moreover we include the last sellout period in the first term. This can be done by assuming that the demand $\tilde{d}_{k_{\max}}$ in the sellout period for $\tilde{n}$ items always amounts to $\tilde{n}$.

We have to show that

$$\sum_{k=0}^{k_{\max}} \tilde{\pi_k} \min\left\{\max\left\{\tilde{n} - \sum_{j=0}^{k-1} \tilde{d}_j, 0\right\}, \tilde{d}_k\right\}$$

is concave for $\tilde{n} > 0$.

We show that $\tilde{\pi}_{f_{\tilde{n_0}}}$ with $f_{\tilde{n_0}} := \min\left\{k \in K : \sum_{j=0}^{k} \tilde{d}_j \geq \tilde{n_0}\right\}$ is a supergradient for each $\tilde{n_0} > 0$, i.e. we show that for every $\tilde{n}, \tilde{n_0} > 0$ it is

$$\sum_{k=0}^{k_{\max}} \tilde{\pi}_k \min\left\{\max\left\{\tilde{n} - \sum_{j=0}^{k-1} \tilde{d}_j, 0\right\}, \tilde{d}_k\right\}$$
$$- \sum_{k=0}^{k_{\max}} \tilde{\pi}_k \min\left\{\max\left\{\tilde{n_0} - \sum_{j=0}^{k-1} \tilde{d}_j, 0\right\}, \tilde{d}_k\right\} \leq \tilde{\pi}_{f_{\tilde{n_0}}}\left(\tilde{n} - \tilde{n_0}\right).$$

Then concavity follows.[1]

It is $f_{\tilde{n_0}}$ as defined above and analogously $f_{\tilde{n}} := \min\{k \in K : \sum_{j=0}^{k} \tilde{d}_j \geq \tilde{n}\}$.

Then it is

$$\sum_{k=0}^{k_{\max}} \tilde{\pi_k} \min\left\{\max\left\{\tilde{n} - \sum_{j=0}^{k-1} \tilde{d}_j, 0\right\}, \tilde{d}_k\right\}$$
$$- \sum_{k=0}^{k_{\max}} \tilde{\pi_k} \min\left\{\max\left\{\tilde{n_0} - \sum_{j=0}^{k-1} \tilde{d}_j, 0\right\}, \tilde{d}_k\right\}$$
$$= \sum_{k=0}^{f_{\tilde{n}}-1} \tilde{\pi_k}\tilde{d_k} + \tilde{\pi}_{f_{\tilde{n}}}\left(\tilde{n} - \sum_{k=0}^{f_{\tilde{n}-1}} d_k\right) - \sum_{k=0}^{f_{\tilde{n_0}}-1} \tilde{\pi_k}\tilde{d_k} - \tilde{\pi}_{f_{\tilde{n_0}}}\left(\tilde{n_0} - \sum_{k=0}^{f_{\tilde{n_0}-1}} d_k\right)$$

[1] For detailed information about supergradients and generalized concavity we refer the reader to [ADSZ10].

If $\tilde{n} = \tilde{n_0}$ the term above takes value 0 and the supergradient inequality follows. So in the following we only consider the two cases $\tilde{n} > \tilde{n_0}$ and $\tilde{n} < \tilde{n_0}$.

Case 1: $\tilde{n} > \tilde{n_0}$

It is $f_{\tilde{n}} \geq f_{\tilde{n_0}}$ and because $\pi_k \geq \pi_{k+1},\ \forall k \in K \setminus \{k_{\max}\}$ it is also $\pi_{f_{\tilde{n}}} \leq \pi_{f_{\tilde{n_0}}}$. Such it is

$$
\begin{aligned}
&\sum_{k=0}^{f_{\tilde{n}}-1} \tilde{\pi_k}\tilde{d_k} + \tilde{\pi}_{f_{\tilde{n}}}(\tilde{n} - \sum_{k=0}^{f_{\tilde{n}}-1} \tilde{d_k}) - \sum_{k=0}^{f_{\tilde{n_0}}-1} \tilde{\pi_k}\tilde{d_k} - \tilde{\pi}_{f_{\tilde{n_0}}}(\tilde{n_0} - \sum_{k=0}^{f_{\tilde{n_0}}-1} \tilde{d_k}) \\
=&\sum_{k=f_{\tilde{n_0}}}^{f_{\tilde{n}}-1} \tilde{\pi_k}\tilde{d_k} + \tilde{\pi}_{f_{\tilde{n}}}\tilde{n} - \tilde{\pi}_{f_{\tilde{n}}} \sum_{k=0}^{f_{\tilde{n}}-1} \tilde{d_k} - \tilde{\pi}_{f_{\tilde{n_0}}}\tilde{n_0} + \tilde{\pi}_{f_{\tilde{n_0}}} \sum_{k=0}^{f_{\tilde{n_0}}-1} \tilde{d_k} \\
\leq&\tilde{\pi}_{f_{\tilde{n_0}}} \sum_{k=f_{\tilde{n_0}}}^{f_{\tilde{n}}-1} \tilde{d_k} + \tilde{\pi}_{f_{\tilde{n}}}\tilde{n} - \tilde{\pi}_{f_{\tilde{n}}} \sum_{k=0}^{f_{\tilde{n}}-1} \tilde{d_k} - \tilde{\pi}_{f_{\tilde{n_0}}}\tilde{n_0} + \tilde{\pi}_{f_{\tilde{n_0}}} \sum_{k=0}^{f_{\tilde{n_0}}-1} \tilde{d_k} \\
=&\tilde{\pi}_{f_{\tilde{n_0}}} \sum_{k=0}^{f_{\tilde{n}}-1} \tilde{d_k} + \tilde{\pi}_{f_{\tilde{n}}}\tilde{n} - \tilde{\pi}_{f_{\tilde{n}}} \sum_{k=0}^{f_{\tilde{n}}-1} \tilde{d_k} - \tilde{\pi}_{f_{\tilde{n_0}}}\tilde{n_0} \\
=&\left(\tilde{\pi}_{f_{\tilde{n_0}}} - \tilde{\pi}_{f_{\tilde{n}}}\right) \sum_{k=0}^{f_{\tilde{n}}-1} \tilde{d_k} + \tilde{\pi}_{f_{\tilde{n}}}\tilde{n} - \tilde{\pi}_{f_{\tilde{n_0}}}\tilde{n_0} \\
\leq&\left(\tilde{\pi}_{f_{\tilde{n_0}}} - \tilde{\pi}_{f_{\tilde{n}}}\right) \tilde{n} + \tilde{\pi}_{f_{\tilde{n}}}\tilde{n} - \tilde{\pi}_{f_{\tilde{n_0}}}\tilde{n_0} = \tilde{\pi}_{f_{\tilde{n_0}}}(\tilde{n} - \tilde{n_0}).
\end{aligned}
$$

Case 2: $\tilde{n} < \tilde{n_0}$

It is $f_{\tilde{n}} \leq f_{\tilde{n_0}}$ and because $\pi_k \leq \pi_{k+1},\ \forall k \in K \setminus \{k_{\max}\}$ it is also $\pi_{f_{\tilde{n}}} \geq \pi_{f_{\tilde{n_0}}}$. Such it is

$$
\begin{aligned}
&\sum_{k=0}^{f_{\tilde{n}}-1} \tilde{\pi_k}\tilde{d_k} + \tilde{\pi}_{f_{\tilde{n}}}(\tilde{n} - \sum_{k=0}^{f_{\tilde{n}}-1} \tilde{d_k}) - \sum_{k=0}^{f_{\tilde{n_0}}-1} \tilde{\pi_k}\tilde{d_k} - \tilde{\pi}_{f_{\tilde{n_0}}}(\tilde{n_0} - \sum_{k=0}^{f_{\tilde{n_0}}-1} \tilde{d_k}) \\
=&\sum_{k=0}^{f_{\tilde{n}}-1} (\pi_k - \pi_{f_{\tilde{n}}})d_k + \pi_{f_{\tilde{n}}}\tilde{n} + \sum_{k=0}^{f_{\tilde{n_0}-1}} (\pi_{f_{\tilde{n_0}}} - \pi_k)d_k - \pi_{f_{\tilde{n_0}}}\tilde{n_0} \\
=&\sum_{k=0}^{f_{\tilde{n}}} (\pi_{f_{\tilde{n_0}}} - \pi_{f_{\tilde{n}}})d_k + \sum_{k=f_{\tilde{n}}+1}^{f_{\tilde{n_0}}-1} (\pi_{f_{\tilde{n}_0}} - \pi_k)d_k + \pi_{f_{\tilde{n}}}\tilde{n} - \pi_{f_{\tilde{n_0}}}\tilde{n_0} \\
\leq&\tilde{n}(\pi_{f_{\tilde{n_0}}} - \pi_{f_{\tilde{n}}}) + \pi_{f_{\tilde{n}}}\tilde{n} - \pi_{f_{\tilde{n_0}}}\tilde{n_0} = \pi_{f_{\tilde{n_0}}}(\tilde{n} - \tilde{n_0})
\end{aligned}
$$

□

One possibility to compute the single supply revenues for all possible numbers of supplied items n for a price trajectory and a given scenario would be to consider each n separately and then apply the formula from Definition 14. But this may lead to unnecessary iterations: For all $m < n$ and all periods k with $\sum_{j=1}^{k} d^e_{j,b,s,t_j} \leq m$ the yield in this period as well for m as for n amounts to $\exp(-\rho k)\pi_{t_k} d^e_{k,b,s,t_k}$. With this consideration iterations can be avoided. Nevertheless additional mark-down costs may arise for a higher supply. To handle these we define the *aggregated discount*.

Definition 15 (aggregated discount)**.** We define the *aggregated discount* $\tilde{\rho}_k^t$ for Period k with $0 < k < k_{\max}$ and a price trajectory $t = (t_0, \ldots, t_{k_{\max}})$ as

$$\tilde{\rho}_k^t := \sum_{j \in K_k^t} \exp(-\rho j) \tag{7.2}$$

while $j \in K_k^t$ if and only if $0 < j \leq k$ and $t_j \neq t_{j-1}$.

The aggregated discount arises by adding up the discounting factors for all real sales periods $\leq k$ in which a mark-down occurs.

Algorithm 6 considers different numbers of supplied items simultaneously when traversing the sales periods.

Algorithm 6 Single supply revenue

Require: complete data of an instance of ISPO,
map $e \to t$ from Scenario e to Price trajectory $t = (t_0, t_1 \ldots, t_{k_{\max}})$
Ensure: for every branch b, every size s and integer numbers
$n : v^{\min} \leq n \leq \lceil \sum_{k=0}^{k_{\max}-1} d^e_{k,b,s,t_k} \rceil$ single supply revenue $\bar{a}^{e \to t}_{b,s,n}$

1: **for all** $b \in B$ **do**
2: **for all** $s \in S$ **do**
3: set $\tilde{r} = v^{\min}$
4: set $\tilde{a} = -\mathrm{ap} \cdot v^{\min}$
5: set $\tilde{d} = d^e_{0,b,s,t_0}$
6: set $\tilde{k} = 0$
7: set $\tilde{p} = t_0$
8: set $n = v^{\min}$
9: **while** true **do**
10: **if** $\tilde{k} > 0$ **and** $t_{\tilde{k}} \neq t_{\tilde{k}-1}$ **then**
11: $\tilde{a} = \tilde{a} - \exp\left(-\rho\tilde{k}\right) \tilde{r}\mu_v$
12: **end if**
13: **if** $\tilde{d} < \tilde{r}$ **then**
14: $\tilde{a} = \tilde{a} + exp\left(-\rho\tilde{k}\right) \tilde{d}\pi_{\tilde{p}}$
15: $\tilde{r} = \tilde{r} - \tilde{d}$
16: **if** $\tilde{k} < k_{\max} - 2$ **then**
17: $\tilde{k} = \tilde{k} + 1$
18: $\tilde{p} = t_{\tilde{k}}$
19: $\tilde{d} = d^e_{\tilde{k},b,s,t_{\tilde{k}}}$
20: **else**
21: $\bar{a}^{e \to t}_{b,s,n} = \tilde{a} + \exp(-\rho k_{\max}) \left(\tilde{r}\left(\pi_{t_{k_{\max}}} - q_{k_{\max}}\mu_v\right)\right)$
22: **break**
23: **end if**
24: **else**
25: $\tilde{a} = \tilde{a} + \exp\left(-\rho\tilde{k}\right) \tilde{r}\pi_{\tilde{p}}$
26: $\bar{a}^{e \to t}_{b,s,n} = \tilde{a}$
27: $\tilde{d} = \tilde{d} - \tilde{r}$
28: $\tilde{r} = 1$
29: $n = n + 1$
30: $\tilde{a} = \tilde{a} - \mathrm{ap}$
31: **if** $\tilde{k} > 0$ **then**
32: $\tilde{a} = \tilde{a} - \tilde{\rho}^t_{\tilde{k}-1}\mu_v$
33: **end if**
34: **end if**
35: **end while**
36: **end for**
37: **end for**

The proceeding in Algorithm 6 is as follows:

We traverse the branches and sizes. First we set the current stock $\tilde{r}$ for the considered branch and size to the minimum supply per lot-type and size $v^{\min}$, Step 3.

For each period $\tilde{k}$ we sell the minimum of current stock and current demand $\tilde{d}$, Case 13 or 24. The revenue is then computed by Step 14 or Step 25. If there is a mark-down

at the beginning of the current period, we have to subtract the variable mark-down costs in Step 11.

If the demand of the considered period is smaller than the current stock, we have to update the stock, Step 15. We possibly go to the next sales period, Step 17, and after updating demand and price index we continue. If we already are in the sellout period $k_{\max}$ we update the current revenue by adding the salvage value and subtracting the variable mark-down costs for the remaining items in Step 21 and update the revenue for the current number n. After that we break the loop.

In the other case – if the demand is not smaller than the current stock – we will sell all of our current stock and state the revenue for the considered number of supplied items, Step 26. We update the remaining demand for the current period $\tilde{k}$ in Step 27 and increase our stock by one in Step 28 to compute the revenue for the next number of supplied items, Step 29. For the additionally supplied item we have to pay the acquisition price, see Step 30. Moreover, because the additionally supplied item was in the considered branch at the beginning of the sales process – before we got here for the current n the demand was always smaller than the current stock $\tilde{r}$ – we have to subtract variable mark-down costs for all former periods, Step 11. We continue with the updated number n.

7.1.1 Runtime of Algorithm 6

In the following we want to outline results in terms of the runtime of Algorithm 6.

Theorem 6 (Number of iterations in Algorithm 6)**.** *We consider a branch $b \in B$, a size $s \in S$ and a map scenario to price trajectory $e \to t$ with $t = (t_0, \dots, t_{k_{\max}})$. With d_k we denote the demand d^e_{k,b,s,t_k} for period k according to t. For $v^{\min} = 1$ the number of iterations $it^{(1)}$ in the while-loop of Algorithm 6 is given by*

$$it^{(1)} = k_{\max} + \left\lfloor \sum_{k=0}^{k_{\max}-1} d_k \right\rfloor. \tag{7.3}$$

To proof this theorem we exploit the following lemmas:

Lemma 2 (equality condition)**.** *It is $0 \leq \tilde{d} < \tilde{r} \leq 1$. Then the following equation holds:*

$$\tilde{r} - \tilde{d} = 1 - \tilde{d} + \tilde{r} + \lfloor \tilde{d} - \tilde{r} \rfloor. \tag{7.4}$$

Proof. Since $0 \leq \tilde{d} < \tilde{r} \leq 1$ it is $\tilde{d} - \tilde{r} < 0$ and $\tilde{d} - \tilde{r} \geq -1$. This means $\lfloor \tilde{d} - \tilde{r} \rfloor = -1$ and the claim follows. □

Lemma 3 (value of $\tilde{r}$ in Algorithm 6 (part 1))**.** *It is $v^{\min} = 1$. With $\tilde{r}_k$ we denote the value of $\tilde{r}$ at the first iteration of the while-loop for period $\tilde{k} = k$ with $0 \leq k < k_{\max}$. The corresponding demand in period k is given by $d_k := d^e_{k,s,b,t_k}$. It is*

$$\tilde{r}_k = 1 - d_{k-1} + \tilde{r}_{k-1} + \lfloor d_{k-1} - \tilde{r}_{k-1} \rfloor. \tag{7.5}$$

Proof. We consider two cases:
Case 1: $d_{k-1} < \tilde{r}_{k-1}$
With Step 15 of the algorithm it is $\tilde{r}_k = \tilde{r}_{k-1} - d_{k-1}$. Moreover it is $\tilde{r}_k \leq 1$ (Step 3 and Step 28). Applying Lemma 2 provides the equation.
Case 2: $d_{k-1} \geq \tilde{r}_{k-1}$
Step 27 of the algorithm provides $\tilde{d} = d_{k-1} - \tilde{r}_{k-1}$ and Step 28 sets $\tilde{r} = 1$. As long as $\tilde{d} \geq \tilde{r}$ Step 27 yields $\tilde{d} = \tilde{d} - 1$ because $\tilde{r}$ has been set to 1 before in Step 28. That means we have to subtract one time $\tilde{r}_{k-1}$ and then $\lfloor d_{k-1} - \tilde{r}_{\tilde{k}-1} \rfloor$ times $\tilde{r} = 1$ for updating $\tilde{d}$. When finally $\tilde{d} < \tilde{r}$, it is $\tilde{d} = d_{k-1} - \tilde{r}_{k-1} - \lfloor d_{k-1} - \tilde{r}_{k-1} \rfloor$ and after Step 15 it is $\tilde{r}_k = 1 - (d_{k-1} - \tilde{r}_{k-1} - \lfloor d_{k-1} - \tilde{r}_{k-1} \rfloor)$.

□

Lemma 4 (Value of $\tilde{r}$ in Algorithm 6 (part 2))**.** *The requirements and notations of Lemma 3 are met. Then the value of $\tilde{r}_k$ is given by*

$$\tilde{r}_k = 1 - \sum_{j=0}^{k-1} d_j + \left\lfloor \sum_{j=0}^{k-1} d_j \right\rfloor . \tag{7.6}$$

Proof. We use Lemma 3 and prove the equality by induction.
$k = 0$:
The claim is true because the sums of demands take value zero. So $\tilde{r}_0 = 1$, what is true by assumption.
$k \to k + 1$:
We assume that the claim is valid for k. Then with Lemma 3 it is
$\tilde{r}_{k+1} = 1 - d_k + \tilde{r}_k + \lfloor d_k - \tilde{r}_k \rfloor$.
By applying the induction hypothesis we get

$$\begin{aligned}\tilde{r}_{k+1} =& 1 - d_k + 1 - \sum_{j=0}^{k-1} d_j + \left\lfloor \sum_{j=0}^{k-1} d_j \right\rfloor \\ &+ \left\lfloor d_k - \left(1 - \sum_{j=0}^{k-1} d_j + \left\lfloor \sum_{j=0}^{k-1} d_j \right\rfloor \right) \right\rfloor .\end{aligned}$$

By extracting the integer values from rounding we finally get

$$\tilde{r}_{k+1} = 1 - \sum_{j=0}^{k} d_j + \left\lfloor \sum_{j=0}^{k} d_j \right\rfloor .$$

□

For a period $k < k_{\max}$ the number of iterations needed in the while-loop of Algorithm 6 depends on the current stock and demand. If the demand is greater than or equals the stock then we are first in the case of Step 24. We subtract the current demand from our stock and set the current stock to one. Altogether we need one iteration for subtracting the demand from the current stock and then have to set our current stock $\lfloor d_k - \tilde{r}_k \rfloor$ times to one, Step 15, to meet the demand of Period k. We need another iteration in which we increase the current period by one, Step 17.

If the demand is smaller than the current stock we only perform one iteration in the while loop: the period is increased by one in Step 17.

Observation 1 (Number of iterations for $\tilde{k}$ in Algorithm 6)**.** *It is* $v^{\min} = 1$*. For period* k *with* $0 \leq k < k_{\max}$ *the number of iterations* $it_k^{(1)}$ *while* $\tilde{k} = k$ *in the while-loop in Algorithm 6 is*

$$it_k^{(1)} = 2 + \lfloor d_k - \tilde{r}_k \rfloor. \tag{7.7}$$

With the help of the previous lemmas and Observation 1 we now prove Theorem 6.

Proof of Theorem 6. We first put the results of Lemma 4 and Observation 1 together and get

$$it_k^{(1)} = 2 + \left\lfloor d_k - 1 + \sum_{j=0}^{k-1} d_j - \left\lfloor \sum_{j=0}^{k-1} d_j \right\rfloor \right\rfloor.$$

Extracting the integer values -1 and $\lfloor \sum_{j=0}^{k-1} d_j \rfloor$ from rounding yields

$$it_k^{(1)} = 1 + \left\lfloor \sum_{j=0}^{k} d_j \right\rfloor - \left\lfloor \sum_{j=0}^{k-1} d_j \right\rfloor.$$

Summing up over the real sales periods yields

$$\begin{aligned} it^{(1)} =& \sum_{k=0}^{k_{\max}-1} \left(1 + \left\lfloor \sum_{j=0}^{k} d_j \right\rfloor - \left\lfloor \sum_{j=0}^{k-1} d_j \right\rfloor \right) \\ =& k_{\max} + \sum_{k=0}^{k_{\max}-1} \left\lfloor \sum_{j=0}^{k} d_j \right\rfloor - \sum_{j=0}^{k_{\max}-1} \left\lfloor \sum_{j=0}^{k-1} d_j \right\rfloor \\ =& k_{\max} + \left\lfloor \sum_{k=0}^{k_{\max}-1} d_k \right\rfloor. \end{aligned}$$

□

If $v^{\min} > 1$ the runtime of Algorithm 6 will be lower or equal.

Corollary 6 (number of iterations for Algorithm 6 with $v^{\min} \geq 1$)**.** *The general number of iterations* $it^{(v^{\min})}$ *in the while-loop for the minimum number of supplied items per branch and size* $v^{\min}$ *is given by*

$$it^{(v^{\min})} = 1 - v_{\min} + k_{\max} + \max \left\{ \left\lfloor \sum_{k=0}^{k_{\max}-1} d_k \right\rfloor, v_{\min} - 1 \right\} \tag{7.8}$$

Proof. If $v^{\min} = 1$, the maximum in the last term of 7.8 amounts to $\left\lfloor \sum_{k=0}^{k_{\max}-1} d_k \right\rfloor$ and the claim follows directly.

In the case of $v^{\min} > 1$ we first assume that there exists a period f with

$$f = \min \left\{ k \in \{0, \ldots, k_{\max} - 1\} : \sum_{j=0}^{k} d_k \geq v^{\min} \right\}.$$

We will show that in this case

$$it^{(v^{\min})} = \underbrace{f}_{:=it^{<}} + \underbrace{2 + \left\lfloor \sum_{k=0}^{f} d_k \right\rfloor - v^{\min}}_{:=it^{=}} + \underbrace{k_{\max} - f - 1 + \left\lfloor \sum_{k=0}^{k_{\max}-1} d_k \right\rfloor - \left\lfloor \sum_{k=0}^{f} d_k \right\rfloor}_{:=it^{>}}.$$

By simplification then the claim follows.

While $\sum_{k=0}^{\tilde{k}} d_k < v^{\min}$ we only do one iteration for the current period $\tilde{k}$ and then change to the next period. This results in $it^{<} = f$ iterations.

The first time when $\sum_{k=0}^{\tilde{k}} d_{\tilde{k}} \geq v^{\min}$, it is $\tilde{r}_{\tilde{k}} = v^{\min} - \sum_{k=0}^{\tilde{k}-1} d_k$. In Step 15 $\tilde{r}$ is set to one. We can apply Observation 1 and get

$$it^{=} = 2 + \left\lfloor d_{\tilde{k}} - v^{\min} + \sum_{k=0}^{\tilde{k}-1} d_k \right\rfloor = 2 + \left\lfloor \sum_{k=0}^{\tilde{k}} d_k \right\rfloor - v^{\min}$$

iterations for $\tilde{k} = f$.

We now will consider the number of iterations for $\tilde{k} > f$. It is

$$\tilde{r}_f = v^{\min} - \sum_{k=0}^{f-1} d_k.$$

Because $\tilde{r}$ is set to value 1 at the end of the first iteration with $\tilde{k} = f + 1$ Lemma 3 holds. Inserting $\tilde{r}_{\tilde{k}-1} = v^{\min} - \sum_{k=0}^{f-1} d_k$ in (7.5), Lemma 3, and summing up for the remaining $k_{\max} - f - 1$ periods yields $it^{>} = k_{\max} - f - 1 + \left\lfloor \sum_{k=0}^{k_{\max}-1} d_k \right\rfloor - \left\lfloor \sum_{k=0}^{f} d_k \right\rfloor$ iterations.

In the case that there exists no $0 \leq f < k_{\max}$ with $\sum_{k=0}^{f} d_k \geq v^{\min}$ the maximum of the last term of 7.8 always amounts to $v^{\min} - 1$. We get $k_{\max}$ iterations, what is correct because in the algorithm always $\tilde{d} < \tilde{r}$ and the current period $\tilde{k}$ is always increased by one. □

We now can state the runtime of Algorithm 6.

Corollary 7 (general runtime of Algorithm 6)**.** *The runtime of Algorithm 6 for the map $e \to t$ depending on the minimum supply per branch and size $v^{\min}$ is given by*

$$\mathcal{O}\left(|B||S|\left(k_{\max} + \left\lfloor \sum_{k=0}^{k_{\max}-1} d^{e}_{k,b,s,t_k} \right\rfloor\right)\right). \tag{7.9}$$

In Algorithm 6 we did not regard the number of supplied items per branch and size that are higher than the rounded up demand over all real sales periods. Because every additional item will not be sold until sellout, the additional revenue is easy to compute: We just have to add the salvage value and subtract acquisition price and mark-down costs for all periods for the additional items.

Observation 2 (revenue for demand-exceeding number of items)**.**
It is $n_{\max} := \left\lceil \sum_{k=0}^{k_{\max}-1} d^e_{k,b,s,t_k} \right\rceil$ *the highest number in Algorithm 6 for which for Scenario* e*, Price Trajectory* $t = (t_0, \ldots, t_{k_{\max}})$*, Branch* b *and Size* s *the single supply revenue* $\bar{a}^{e\to t}_{b,s,n}$ *is computed. For every* $m > n_{\max}$ *the single supply revenue* $\bar{a}^{e\to t}_{b,s,m}$ *is given by*

$$\begin{aligned}\bar{a}^{e\to t}_{b,s,m} =& \bar{a}^{e\to t}_{b,s,n_{\max}} + \\ & (m - n_{\max}) \left(-\mathrm{ap} - \mu_v \tilde{\rho}^t_{k_{\max}-1} + \exp\left(-\rho k_{\max}\right)\left(\pi_{p_{\max}} - q_{k_{\max}}\mu_v\right)\right).\end{aligned} \tag{7.10}$$

7.1.2 An Example

Example 8 (Algorithm 6 – computation of the single supply revenue)**.** We now depict the proceeding of Algorithm 6 on our accompanying example from Section 4.7 for $v_{\min} = 1$. We consider the price trajectory $t = (0, 0, 0, 2, 3)$ for the scenario "normal seller" and compute the single supply revenues for Branch 1 and Size S. We state the demands for each period according to t in the subsequent table.

k	0	1	2	3
$d^e_{k,1,S,t_k}$	2	1.5	1	1.5

At the beginning of Algorithm 6 it is $\tilde{r} = 1$ and $\tilde{a} = -0.5$ (Step 30). The demand $\tilde{d}$ for the current period $\tilde{k} = 0$ takes value 2. Therefore, $\tilde{d} > \tilde{r}$ and the current revenue is given by $\tilde{a} = -0.5 + \exp(-\rho \cdot 0) \cdot 1 \cdot 10.99 = 10.49$ (Step 25). We update the revenue for suppling one item: $\bar{a}^{e\to t}_{1,\mathrm{S},1} = 10.49$ (Step 26). The remaining demand $\tilde{d}$ is updated to 1, and now we consider a supply of $n = 2$ and have to subtract the acquisition price for the additional item from the current revenue: $\tilde{a} = 10.49 - 0.5 = 9.99$ (Steps 27-30).

Now, it is $1 = \tilde{d} = \tilde{r}$, that means again we are in the case that $\tilde{d} \geq \tilde{r}$. We update the current revenue $\tilde{a} = 9.99 + \exp(-0.01 \cdot 0) \cdot 1 \cdot 10.99 = 20.98$ and set the revenue for supplying two items: $\bar{a}^{e\to t}_{1,\mathrm{S},2} = 20.98$. The remaining demand now takes value zero and we consider a supply of $n = 3$. The current revenue is set to $\tilde{a} = 20.98 - 0.5 = 20.48$.

It is $0 = \tilde{d} < \tilde{r} = 1$. Because the remaining demand takes value zero, the current revenue and stock does not change (Steps 14 and 15). We increase $\tilde{k}$ to $\tilde{k} = 1$. The remaining demand now is the demand of Period 1: $\tilde{d} = 1.5$, $\tilde{r}$ still takes value 1. Now it is $1.5 = \tilde{d} \geq \tilde{r} = 1$ and we update the revenue according to the sales: $\tilde{a} = 20.48 + \exp(-0.01 \cdot 1) \cdot 1 \cdot 10.99 = 31.36$. We update the revenue for supplying 3 items to $\bar{a}^{e\to t}_{1,\mathrm{S},3} = 31.36$. The remaining demand amounts to $\tilde{d} = 0.5$ and we consider the supply $n = 4$. So we have to regard the acquisition price for the additional item and subtract it from the current revenue: $\tilde{a} = 31.36 - 0.5 = 30.86$.

Still, the remaining demand is smaller than the current stock: $0.5 = \tilde{d} < \tilde{r} = 1$. The updated revenue amounts to $\tilde{a} = 30.86 + \exp(-0.01 \cdot 1) \cdot 0.5 \cdot 10.99 = 36.30$. The remaining stock is updated to $\tilde{r} = 0.5$ and we increase the current period to $\tilde{k} = 2$.

Now, after the demand is updated it is $1 = \tilde{d} \geq \tilde{r} = 0.5$. We update the current revenue and obtain $\tilde{a} = 36.30 + \exp(-0.01 \cdot 2) \cdot 0.5 \cdot 10.99 = 41.69$. The revenue for supplying 4 item amounts to $\bar{a}^{e\to t}_{1,\mathrm{S},4} = 41.69$. We update the remaining demand: $\tilde{d} = 0.5$, set the remaining stock $\tilde{r} = 1$ and continue with a supply of $n = 5$. With subtracting the acquisition price we get $\tilde{a} = 41.69 - 0.5 = 41.19$.

We are in the case that $0.5 = \tilde{d} < \tilde{r} = 1$. We update the current revenue and get $\tilde{a} = 41.19 + \exp(-0.01 \cdot 2) \cdot 0.5 \cdot 10.99 = 46.58$. The current stock is updated to $\tilde{r} = 0.5$. We continue with period $\tilde{k} = 3$ and set the current demand to the demand of Period 3: $\tilde{d} = 1.5$.

Because there is a mark-down in the current period $\tilde{k} = 3$, we have to regard the variable mark-down costs and get $\tilde{a} = 46.58 - \exp(-0.01 \cdot 3) \cdot 0.5 \cdot 0.1 = 46.53$. Now the current demand exceeds the current stock: It is $1.5 = \tilde{d} \geq \tilde{r} = 0.5$. The updated current revenue amounts to $\tilde{a} = 46.53 + \exp(-0.01 \cdot 3) \cdot 0.5 \cdot 1.99 = 47.49$. We set the revenue for suppling 5 items to $\bar{a}^{e \to t}_{1,\mathrm{S},5} = 47.49$. The remaining demand now is $\tilde{d} = 1$, the remaining stock is set to $\tilde{r} = 1$. We go on with suppling $n = 6$ items and subtract the acquisition price from the current revenue: $\tilde{a} = 47.49 - 0,5 = 46.99$.

Additionally we have to regard the mark-down costs for the current stock. It is $\tilde{a} = 46.99 - \exp(-0.01 \cdot 3) \cdot 0.1 = 46.89$ and $1 = \tilde{d} = \tilde{r}$. The updated revenue is $\tilde{a} = 46.89 + \exp(-0.01 \cdot 3) \cdot 1 \cdot 1.99 = 48.82$. The cost for suppling 6 items is set to $\bar{a}^{e \to t}_{1,\mathrm{S},6} = 48.82$. The remaining demand is $\tilde{d} = 0$. We continue with $n = 7$ and set the remaining stock to $\tilde{r} = 1$. The revenue is updated to $\tilde{a} = 48.82 - 0.5 = 48.32$.

The mark-down costs for one additional item are subtracted from the current revenue. This results in $\tilde{a} = 48.32 - \exp(-0.01 \cdot 3) \cdot 0.1 = 48.22$. It is $0 = \tilde{d} < \tilde{r} = 1$. Because we are in the last real sales period, we are in the case of Step 20. We have to add the salvage value and subtract the variable mark-down costs for period $k_{\max}$ and get $\bar{a}^{e \to t}_{1,\mathrm{S},7} = 48.22 + \exp(-0.01 \cdot 4) \cdot 1 \cdot (0.99 - 2 \cdot 0.1) = 48.98$ and end up the computation for Branch 1 and Size S.

The additional revenue per supplied item for more than 7 supplied items amounts to

$$-0.5 + \exp(-0.01 \cdot 3) \cdot 0.1 + \exp(-0.01 \cdot 4) \cdot (0.99 - 2 \cdot 0.1) = 0.16.$$

7.1.3 Computational results

We compare Algorithm 6 with the more intuitive Algorithm 7 to compute the single supply revenues. In Algorithm 7 for each integer supply of items n per branch and size with $v_{\min} \leq n \leq \lceil \sum_{k=0}^{k_{\max}-1} d^e_{k,t_k,b,s} \rceil$ we always walk through the periods $k = 0, \ldots, k_{\max} - 1$ until all items are sold or we reached the last real sales period. Thus, the alternative algorithm has a runtime of

$$\mathcal{O}\left(|B||S| k_{\max} \left\lceil \sum_{k=0}^{k_{\max}-1} d^e_{k,b,s,t_k} \right\rceil \right). \tag{7.11}$$

That means with Algorithm 6 we can reduce the pseudo-polynomial runtime against the more intuitive Algorithm 7 at least by one factor.

In Table 7.1 the results for comparing Algorithms 6 with Algorithm 7 for a subset from the set of real instances $\mathcal{I}$ (see Appendix E) are stated. We took the supply $I_{b,s}$ from historical transaction data we obtained from our partner. The algorithms were performed consecutively for all three scenarios "low seller", "normal seller" and "high seller". Thus, time and number of iterations refer to all three scenarios. The mean values in the last line refer to the whole set $\mathcal{I}$. The number of iterations "#iter" and the computation time in seconds "time(s)" is stated for Algorithm 6 and the described alternative.

In Algorithm 6 on average the number of iterations is 170 586 less. Let us take a look on the overall computation time. It amounts to 49.00 seconds for our proposed

Algorithm 7 Single supply revenue 2

Require: map $e \to t$ from Scenario e from Price trajectory $t = (t_0, \ldots, t_{k_{\max}})$, complete data of an instance for ISPO

Ensure: for every branch b, every size s and integer numbers $n : v^{\min} \leq n \leq \lceil \sum_{k=0}^{k_{\max}-1} d^e_{k,b,s,t_k} \rceil$ single supply revenue $\bar{a}^{e \to t}_{b,s,n}$

for all $b \in B$ **do**
  for all $s \in S$ **do**
    set $n = v^{\min}$
    while true **do**
      set $\tilde{r} = n$
      set $\tilde{a} = -\mathrm{ap} \cdot n$
      set $\tilde{p} = t_0$
      set $\tilde{k} = 0$
      set $\tilde{d} = d^e_{0,b,s,t_0}$
      while $\tilde{k} < k_{\max}$ **do**
        if $\tilde{k} > 0$ and $t_{\tilde{k}-1} \neq t_{\tilde{k}}$ **then**
          $\tilde{a} = \tilde{a} - exp\left(-\rho\tilde{k}\right)\tilde{r}\mu_v$
          if $\tilde{d} < \tilde{r}$ **then**
            $\tilde{a} = \tilde{a} + exp\left(-\rho\tilde{k}\right)\tilde{d}\pi_{\tilde{p}}$
            $\tilde{r} = \tilde{r} - \tilde{d}$
            $\tilde{k} = \tilde{k} + 1$
            $\tilde{d} = d^e_{\tilde{k},b,s,t_{\tilde{k}}}$
            $\tilde{p} = t_{\tilde{k}}$
          else
            $\tilde{a} = \tilde{a} + \exp\left(-\rho\tilde{k}\right)\tilde{r}\pi_{\tilde{p}}$
            break
          end if
        end if
      end while
      $\tilde{a} = \tilde{a} + exp\left(-\rho k_{\max}\right)\tilde{r}\left(\pi_{p_{\max}} - q_{k_{\max}}\mu_v\right)$
      $n = n + 1$
    end while
  end for
end for

Instance	#iter		time(s)	
	Alg.7	Alg.6	Alg.7	Alg.6
1	328 308	212 244	0.06	0.05
2	328 308	212 244	0.06	0.05
3	389 730	224 119	0.06	0.06
4	365 680	218 762	0.06	0.05
5	365 680	218 762	0.07	0.04
6	365 680	218 762	0.05	0.06
7	365 680	218 762	0.06	0.05
8	355 236	217 681	0.05	0.06
9	366 060	218 278	0.05	0.06
10	333 383	212 806	0.05	0.05
11	274 576	203 267	0.05	0.05
12	274 576	203 267	0.06	0.05
13	314 439	209 643	0.05	0.05
14	361 619	235 900	0.08	0.06
15	447 372	284 027	0.09	0.08
⋮	⋮	⋮	⋮	⋮
∅	496 757	326 171	0.0569	0.0606
∑	280 507 337	427 210 602	49.00	52.15

Table 7.1: Comparison – computation of single supply revenues

Algorithm and 52.15 second for the alternative. In terms of time this is only a marginal improvement. Yet, we proposed an algorithm for the computation of the single supply revenues which always needs less iterations than a more intuitive algorithm.

7.2 Establishing lot-type revenues

To obtain revenues for a map $e \to t$ Scenario e to Price trajectory t for a given lot-type ℓ and multiplicity m one has to add up the according to Section 7.1 computed single supply revenues $\bar{a}^{e\to t}_{b,s,n}$ in the following way.

Definition 16 (lot-type revenue)**.** The *lot-type revenue* $\hat{a}_{b,\ell,m}$ for Branch b, Lot-type ℓ and Multiplicity m for a map $e \to t$ is given as

$$\hat{a}^{e\to t}_{b,\ell,m} := \sum_{s\in S} \bar{a}^{e\to t}_{b,s,m\cdot l_s} - m \cdot \mathrm{pcost}. \tag{7.12}$$

By definition the single supply revenue contains as well acquisition price as yield resulting from fixing Price trajectory t to Scenario e, so also the lot-type revenue does. We included the corresponding pick-costs by subtracting $m \cdot \mathrm{pcost}$.

7.3 Fixing price trajectories in the ISPO – an SLDP

Let us consider a subset E' of the scenarios E . We are given a set of maps $W^{E'}$ where each $e \in E'$ is mapped to a price trajectory t^e. The part of the expected revenue of the ISPO according to the subset E' of scenarios is given by the optimal objective value of the following SLDP.

Problem 7 (SLDP($W^{E'}$))**.**

$$\max \quad \sum_{e\in E'} \mathrm{Prob}\,(e) \left(\sum_{b\in B} \sum_{\ell\in L} \sum_{m\in M} \hat{a}^{e\to t^e}_{b,\ell,m} x_{b,\ell,m} - \mu_f \tilde{\delta}^{t^e}_{k_{\max}-1} - \sum_{i=1}^{\kappa} \delta_i \cdot z_i \right) \tag{7.13}$$

subject to

$$\sum_{\ell\in L} \sum_{m\in M} x_{b,\ell,m} = 1 \quad \forall b \in B, \tag{7.14}$$

$$\sum_{m\in M} x_{b,\ell,m} \le y_\ell \quad \forall b \in B, \ell \in L, \tag{7.15}$$

$$\sum_{\ell\in L} y_\ell \le \sum_{i=1}^{\kappa} z_i, \tag{7.16}$$

$$z_i \le z_{i-1} \quad i = 1 \ldots, \kappa, \tag{7.17}$$

$$I_{b,s} = \sum_{\ell\in L} \sum_{m\in M} m \cdot \ell_s \cdot x_{b,\ell,m}, \quad \forall b \in B, s \in S, \tag{7.18}$$

$$I = \sum_{b\in B} \sum_{s\in S} I_{b,s}, \tag{7.19}$$

$$I \in [\underline{I}, \overline{I}], \tag{7.20}$$
$$x_{b,\ell,m} \in \{0,1\} \quad \forall b \in B, \ell \in L, m \in M, \tag{7.21}$$
$$y_\ell \in \{0,1\} \quad \forall \ell \in L, \tag{7.22}$$
$$z_i \in \{0,1\} \quad i = 1, \ldots, \kappa. \tag{7.23}$$

The constraints do not differ from the formulation of Problem 2. Now we want to consider revenue not costs and changed the objective from minimization to maximization. The objective coefficient $\hat{a}_{b,\ell,m}^{e \to t_e}$ is the expected revenue for Scenario e for supplying Branch b with Lot-type ℓ in Multiplicity m excluding the fixed mark-down costs. We have to subtract these costs for the periods with mark-downs separately. This is done by using the aggregated discount from Definition 15 for every scenario. Additionaly we subtract the lot-opening costs in the objective.

Notation 2 (optimal objective value of SLDP$\left(W^{E'}\right)$)**.** With $z^*_{\text{SLDP}(W^{E'})}$ we denote the optimal objective function value of Problem 7.

Notation 3 (optimal objective value of the LP relaxation of SLDP$\left(W^{E'}\right)$)**.** With $z^*_{\text{SLDP-LP}(W^{E'})}$ we denote the optimal objective value of the LP relaxation of the SLDP$\left(W^{E'}\right)$.

7.4 Solving the ISPO by enumerating SLDPs

With the results of the previous sections, theoretically we can solve ISPO by solving an SLDP for every set of maps "scenario to price trajectory" in which each considered scenario is assigned to a price trajectory. The subsequent Corollary follows directly from Theorem 2.

Corollary 8 (solving the ISPO by enumerating SLDPs)**.** *We consider ISPO how it is stated in the formulation of Problem 6. We can solve ISPO by enumerating*

$$\binom{k_{\max} - k_{\text{obs}} + p_{\max} - 1}{p_{\max} - 1}^{|E|}$$

SLDP(W^E)s.

We will illustrate the proceeding on a small example.

Example 9 (ISPO – enumeration tree)**.** For simplicity we assume in this example $k_{\text{obs}} = 2$, $k_{\max} = 3$ and $p_{\max} = 2$. According to Theorem 2 this leads to 2 different price trajectories: $t_0 = (0,0,0,2)$ and $t_1 = (0,0,1,2)$. It is $|E| = 3$. In Figure 7.1 we see the corresponding enumeration tree. A depth corresponds to a scenario $e \in E$. The width corresponds to a price trajectory. At the leaves every scenario is fixed to a price trajectory and solving the corresponding SLDP(W^E) yields a lower bound for the optimal solution of the ISPO. The optimal solution of the ISPO is the optimal solution of the SLDP(W^E) with the highest optimal solution value among all possible set of maps W^E.

For example, for the set of ordered scenarios $E = \{e_1, e_2, e_3\}$ and the set of ordered price trajectories $\{t_1, t_2\}$ at the leaf of the third branch in the enumeration tree in Figure 7.1 we would solve an SLDP(W^E) with $W^E = \{e_1 \to t_1, e_2 \to t_2, e_3 \to t_1\}$.

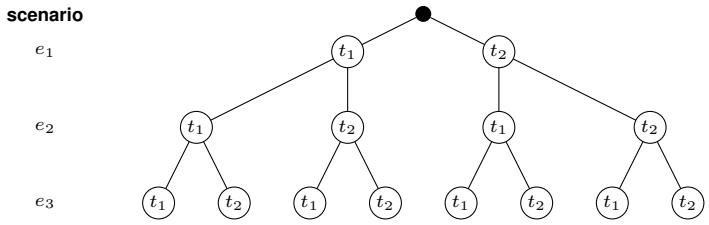

Figure 7.1: ISPO – enumeration tree

Enumerating all maps W^E and solving the related SLDP(W^E) is for us firstly theoretically interesting. With real-world instances with $k_{\text{obs}} = 2$, $k_{\max} = 13$ and $p_{\max} = 4$ this would yield 364 price trajectories for every scenario what means $364^3 = 48\,228\,544$ leaves or to be solved SLDP(W^E)s. Even if we assumed that the solving process for each SLDP(W^E) would take only 1 second, we would have to wait more than 558 days for an optimal solution of the ISPO.

But with this reduction we are able to reduce the size of ISPO to the size of – although – many SLDP(W^E)s and theoretically can solve the same instances – in terms of the numbers of branches, lot-types and multiplicities – as we can handle with the SLDP.

Because also the SLDP(W^E) can be reduced to κ LDPs as shown in Chapter 2, we can use all known approaches for the LDP in the solving process.

Enumerating the SLDP(W^E)s in the outlined way is the basic idea of our exact Branch&Bound solver we will present in Chapter 9. Because solving all SLDP(W^E)s is not possible, we have to prune the enumeration tree. This is done by dual bounds which are topic of the following chapter.

Chapter 8

Dual Bounds

In Chapter 9 we will present an exact Branch&Bound approach to solve the Integrated Size and Price Optimization Problem. One of the most important points in terms of efficiency of this algorithm is the computation of possibly tight dual bounds.

At the nodes of our Branch&Bound we will fix price-trajectories to scenarios. At the leaves then for every scenario a price trajectory is fixed and the corresponding SLDP(W^E) is solved to optimality in order to obtain a primal bound for ISPO. To reduce the number of processed nodes and especially of processed leaves, i.e. to be solved SLDP(W^E)s, we will use lower bounds based on the wait-and-see solution from stochastic programming, see also Chapter 5. We developed tighter dual bounds based on the wait-and-see solution by combining subsets of scenarios and call the result the *extended wait-and-see solution*. We present these bounds in Section 8.1

Computational effort is reduced by combining wait-and-see and extended wait-and-see solutions with relaxations of the SLDP($W^{E'}$) for subsets $E' \subseteq E$ of scenarios. A relaxation we use neglects the fact that the supply has to be in lot-types – i.e. independent integer supplies per branch and size are allowed. Additionally we use LP relaxations. In Section 8.2 we apply these dual bounds to ISPO. We conclude the chapter in Section 8.3.

8.1 Dual bounds from wait-and-see solutions

A well known dual bound from stochastic programming is the wait-and-see solution, see also Chapter 5. In Subsection 8.1.1 we will formulate the wait-and-see solution for a general stochastic mixed-integer program. We show how to extend this bound by combining scenarios, Subsection 8.1.2. Computation time can be reduced by relaxing the underlying problems, Subsection 8.1.3.

8.1.1 The wait-and-see solution

In this section we will refer to the following formulation of a general stochastic mixed-integer program:

Problem 8 (general two-stage SMIP)**.**

$$\max_x z(x(\Xi), \Xi) = \max_x c^T x + \mathbb{E}_{\xi \in \Xi} Q(x, \xi) \tag{8.1}$$

$$\text{subject to } Ax = b, \tag{8.2}$$

$$x \in \mathbb{R}^k \times \mathbb{Z}^{n-k}. \tag{8.3}$$

Ξ is the set of all possible scenarios and $Q(x, \xi)$ is the objective of the second stage for the particular scenario ξ and may be nonlinear.

With $x^*(\Xi)$ we denote the optimal solution of the problem, $z(x^*(\Xi), \Xi)$ is the optimal objective value. If we restrict Problem 8 to a particular scenario ξ we obtain Problem 9.

Problem 9 (general two-stage SMIP associated with Scenario ξ)**.**

$$\max_x z^{\text{SIN}}(x(\xi), \xi) = \max_x c^T x + Q(x, \xi) \tag{8.4}$$

$$\text{subject to } Ax = b, \tag{8.5}$$

$$x \in \mathbb{R}^k \times \mathbb{Z}^{n-k}. \tag{8.6}$$

With $x^*(\xi)$ we denote the optimal solution of Problem 9. The related optimal objective value is denoted by $z^{\text{SIN}}(x^*(\xi), \xi)$.

Definition 17 (wait-and-see solution for a general SMIP)**.** The wait-and-see solution WS for Problem 8 is given by

$$WS = \mathbb{E}_{\xi \in \Xi} \left[\max_x z^{\text{SIN}}(x(\xi), \xi) \right] = \mathbb{E}_{\xi \in \Xi} z^{\text{SIN}}(x^*(\xi), \xi). \tag{8.7}$$

The wait-and-see solution is a dual bound of the original problem. It follows from the proof of Theorem 4 in Chapter 5. Integrality of variables does not affect the result.

8.1.2 Extending wait-and-see solutions

Based on the wait-and-see solution from the previous subsection we developed stronger dual bounds for general two-stage stochastic optimization problems by considering subsets of scenarios. We call them extended wait-and-see solutions.

For this purpose we define the so-called partial stochastic program on which the extended wait-and-see solution is based.

Problem 10 (partial two-stage SMIP)**.** *It is p_ξ the probability of the occurrence of scenario ξ. The partial two-stage stochastic (mixed-integer) program for the subset of scenarios $\Xi' \subseteq \Xi$ is given by*

$$\max_x z^{\text{PA}}(x(\Xi'), \Xi') = \max_x \sum_{\xi \in \Xi'} p_\xi (c^T x + Q(x, \xi)) \tag{8.8}$$

$$\textit{subject to } Ax = b, \tag{8.9}$$

$$x \geq 0. \tag{8.10}$$

It is $x^*(\Xi')$ the optimal solution of the problem and with $z^{\text{PA}}(x^*(\Xi'), \Xi')$ we denote the corresponding objective value.

Definition 18 (extended wait-and-see solution)**.** We define the extended wait-and-see solution $\tilde{WS}(\Xi')$ of an SMIP for the subset $\Xi' \subseteq \Xi$ of scenarios by

$$\tilde{WS}(\Xi') := z^{\mathrm{PA}}(x^*(\Xi'), \Xi') + \sum_{\xi \in \Xi \setminus \Xi'} z^{\mathrm{PA}}(x^*(\{\xi\}), \{\xi\}) \tag{8.11}$$

$$= z^{\mathrm{PA}}(x^*(\Xi'), \Xi') + \sum_{\xi \in \Xi \setminus \Xi'} p_\xi z^{\mathrm{SIN}}(x^*(\xi), \xi). \tag{8.12}$$

For every subset of scenarios from the set Ξ the extended wait-and-see solution is a tighter dual bound than the classical wait-and-see solution.

Theorem 7 (extended wait-and-see solution as dual bound)**.** *For the optimal solution of Problem 8 and the extended wait-and-wee solution $\tilde{WS}(\Xi')$ the following inequalities hold:*

$$WS \geq \tilde{WS}(\Xi') \geq z(x^*(\Xi), \Xi). \tag{8.13}$$

Proof. The first inequality equals

$$\sum_{\xi \in \Xi} p_\xi z^{\mathrm{SIN}}(x^*(\xi), \xi) \geq z^{\mathrm{PA}}(x^*(\Xi'), \Xi') + \sum_{\xi \in \Xi \setminus \Xi'} p_\xi z^{\mathrm{SIN}}(x^*(\xi), \xi).$$

Subtracting $\sum_{\xi \in \Xi \setminus \Xi'} z^{\mathrm{SIN}}(x^*(\xi), \xi)$ on both sides yields

$$\sum_{\xi \in \Xi'} p_\xi z^{\mathrm{SIN}}(x^*(\xi), \xi) \geq z^{\mathrm{PA}}(x^*(\Xi'), \Xi').$$

With $z^{\mathrm{PA}}(x^*(\Xi'), \xi)$ we denote the objective value that results from applying the optimal solution of Problem 10 to Problem 9. By definition it is

$$z^{\mathrm{SIN}}(x^*(\xi), \xi) \geq z^{\mathrm{PA}}(x^*(\Xi'), \xi)$$

for every scenario $\xi \in \Xi'$. Multiplying both sides with the corresponding scenario probability and adding up for all scenarios $\xi \in \Xi'$ yields the claim.

The proof of the second inequality is similar. We denote with $z^{\mathrm{SIN}}(x^*(\Xi), \xi)$ and $z^{\mathrm{PA}}(x^*(\Xi), \Xi')$ the objective values that result from applying $x^*(\Xi)$ to the problems 9 and 10. By definition it is

$$z^{\mathrm{SIN}}(x^*(\xi), \xi) \geq z^{\mathrm{SIN}}(x^*(\Xi), \xi)$$

and also by definition it is

$$z^{\mathrm{PA}}(x^*(\Xi'), \Xi') \geq z^{\mathrm{PA}}(x^*(\Xi), \Xi').$$

Multiplying the first inequality for each scenario $\xi \in \Xi \setminus \Xi'$ on both sides with the corresponding scenario probability does not change the relation. Adding up the second inequality and the with the scenario probabilities multiplied first inequalities for all scenarios $\xi \in \Xi \setminus \Xi'$ yields the claim. □

Partial SMIPs and group subproblems

In Chapter 5 we introduced the so-called group subproblem as it is defined by Sandikçi et.al. [SKS12]. The group subproblem yields a hierarchy of bounds for stochastic

programs by combining all subsets of scenarios with same cardinality and computing the expected optimal value over all group subproblems for these subsets.

We recognized similarities between the group subproblem and our extended wait-and-wee solution. To outline our ideas we first consider a group subproblem where the reference scenario ξ_r takes probability zero, i.e. the reference scenario is not contained in our subset Ξ of scenarios.

For our formulation of a general SMIP, Problem 8, the group subproblem for a subset Ξ' of scenarios can be stated as follows.

Problem 11 (group subproblem for a general SMIP, reference scenario not in Ξ)**.**

$$z_{\mathrm{g}}^*(\Xi') = \max_x c^T x + \sum_{\xi \in \Xi'} \frac{p_\xi}{\rho(\Xi')} Q(x, \xi) \tag{8.14}$$

$$\text{subject to } Ax = b, \tag{8.15}$$

$$x \in \mathbb{R}^k \times \mathbb{Z}^{n-k}. \tag{8.16}$$

where $\rho(\Xi') := \sum_{\xi \in \Xi'} p_\xi$.

It is

$$\rho(\Xi') \cdot z_{\mathrm{g}}^*(\Xi') = \max_x \rho(\Xi') c^T x + \rho(\Xi') \sum_{\xi \in \Xi'} \frac{p_\xi}{\rho(\Xi')} Q(x, \xi) \tag{8.17}$$

$$= \max_x \rho(\Xi') c^T x + \sum_{\xi \in \Xi'} p_\xi Q(x, \xi) \tag{8.18}$$

$$= \max_x \sum_{\xi \in \Xi'} p_\xi (c^T x + Q(x, \xi)). \tag{8.19}$$

This is exactly the objective function of our partial two-stage SMIP, Problem 10.

That means we can compute the expected value of the group subproblem for i scenarios, with $\mathcal{P}_i(\Xi)$ being the set of all subsets of i scenarios from the set of all scenarios Ξ, in this case originally given by

$$EGSO(i) = \frac{1}{\binom{|\Xi| - 1}{i - 1}} \sum_{\Xi' \in \mathcal{P}_i(\Xi)} \rho(\Xi') z_{\mathrm{g}}^*(\Xi') \tag{8.20}$$

by

$$EGSO(i) = \frac{1}{\binom{|\Xi| - 1}{i - 1}} \sum_{\Xi' \in \mathcal{P}_i(\Xi)} z^{\mathrm{PA}}(x^*(\Xi'), \Xi'). \tag{8.21}$$

Our partial two-stage SMIP yields the same optimal solution as the group subproblem. The objective differs because in relation to the formulation of the group subproblem we scaled the vector c by the summed up scenario probabilities.

Such, our partial SMIP can be seen as a special case of the group subproblem where the reference scenario is not contained in the set Ξ.

The wait-and-see solution is with $\Xi' = \emptyset$ a special case of the extended wait-and-see solution. Therefore, in the following we will refer solely to extended wait-and-see solutions.

8.1.3 Relaxations of extended wait-and-see solutions

If the original problem restricted to one scenario is still hard to solve, we can use the bounding property of the wait-and-see solution together with the bounding property of relaxations and use the optimal values of the relaxed problems 9 and 10 for the computation of the extended wait-and-see solutions. The inequalities from the theorems 4 and 7 will also hold for the relaxed extended wait-and-see solutions, the bounds at most increase.

Definition 19 (relaxed extended wait-and-see solution)**.** It is $x_R^*(\xi)$ the optimal solution of a relaxation of Problem 9. With $z_R^{\mathrm{SIN}}(x_R^*(\xi), \xi)$ we denote the related optimal objective value. With $x_R^*(\Xi')$ we denote the optimal solution of a relaxation for Problem 10. The related optimal objective value is $z_R^{\mathrm{PA}}(x_R^*(\Xi'), \Xi')$. We define the *relaxed extended wait-and-see solution* $\tilde{WS}_R(\Xi')$ for the subset Ξ' of scenarios as follows:

$$\tilde{WS}_R(\Xi') := z_R^{\mathrm{PA}}(x_R^*(\Xi'), \Xi') + \sum_{\xi \in \Xi \setminus \Xi'} z_R^{\mathrm{PA}}(x_R^*(\{\xi\}), \{\xi\}) \tag{8.22}$$

$$= z_R^{\mathrm{PA}}(x_R^*(\Xi'), \Xi') + \sum_{\xi \in \Xi \setminus \Xi'} p_\xi z_R^{\mathrm{SIN}}(x_R^*(\xi), \xi) \tag{8.23}$$

Corollary 9 (bounding by relaxed extended wait-and-see solutions)**.** *It is*

$$\tilde{WS}_R(\Xi') \geq \tilde{WS}(\Xi') \geq z(x^*(\Xi), \Xi) \tag{8.24}$$

for all subsets of scenarios $\Xi' \subseteq \Xi$*.*

Proof. The theorem follows from Theorem 7 by exploiting the dual-bound property of relaxations. □

8.2 Application to ISPO

Now we apply the outlined bounds to the ISPO. Because the SLDP in general is hard to solve in our solver we solely use relaxed (extended) wait-and-see solutions. On the one side LP relaxations and on the other side combinatorial bounds – based on relaxing the restriction that the items have to be supplied in terms of lot-types – are applied. In Subsection 8.2.1 we present our combinatorial bounds. Then, in Subsection 8.2.2 we apply the different dual bounds to ISPO and in Subsection 8.2.3 we present computational results.

8.2.1 Relaxing the lot-type constraint – single supply relaxations

By neglecting the restriction that the supply has to be in terms of lots and allowing independent integer supplies of items among branches and sizes we obtain a relaxation of the SLDP($W^{E'}$).

First we will present the problem formulation of this relaxation as an integer linear program before we outline a fast greedy algorithm to solve it.

Problem formulation

The *single supply relaxation* of Problem 7 can be formulated as follows:

Problem 12 (SLDP-CB($W^{E'}$))**.**

$$\max \quad \sum_{e\in\tilde{E}} \text{Prob}(e)\Bigg(\sum_{b\in B}\sum_{s\in S}\sum_{n\in N} \bar{a}^{e\to t}_{b,s,n} x_{b,s,n} - \mu_f \tilde{\delta}^{t^e}_{k_{\max}-1} - |B|\text{pcost} - \delta_1\Bigg) \tag{8.25}$$

$$\text{subject to} \quad \sum_{n\in N} x_{b,s,n} = 1 \quad \forall b \in B, \forall s \in S, \tag{8.26}$$

$$\tilde{vl}^{\min} \leq \sum_{s\in S}\sum_{n\in N} n \cdot x_{b,s,n} \leq \tilde{vl}^{\max} \quad \forall b \in B, \tag{8.27}$$

$$I = \sum_{b\in B}\sum_{s\in S}\sum_{n\in N} n \cdot x_{b,s,n}, \tag{8.28}$$

$$I \in [\underline{I}, \bar{I}], \tag{8.29}$$

$$x_{b,s,n} \in \{0,1\} \ \forall b \in B, s \in S, n \in N. \tag{8.30}$$

The binary variable $x_{b,s,n}$ indicates if Branch b is supplied by n items of Size s. If this is the case it takes value one, and zero otherwise. With Constraint (8.26) it is ensured that exactly one number n from a set of positive integer values N of possibly supplied items is assigned to every branch and size. With Constraint (8.27) given minimum and maximum supplies per branch $\tilde{vl}^{\min}$ and $\tilde{vl}^{\max}$ are adhered to. Abidance of lower and upper bounds for the overall supply is preserved by (8.28) and (8.29). The implicitly integer variable I here denotes the supply over all branches and sizes.

Since in the original formulation of the SLDP every branch is at least supplied by a number of lot-types given by the minimum multiplicity 1 and at least one lot-type has to be used for all branches, we subtract the corresponding pick and lot-opening costs from the objective to strengthen the bound and also regard the fixed mark-down costs for the mapped price trajectories (8.25).

Notation 4 (optimal objective value of SLDP-CB($W^{E'}$)**.** We denote the optimal objective value of Problem 12 with $z^*_{\text{SLDP-CB}(W^{E'})}$.

Observation 3 (possible single supplies)**.** *The set of* possible single supplies N *in the formulation of Problem 12 can be restricted to*

$$N := \{\tilde{v}^{\min}, \tilde{v}^{\min}+1, \tilde{v}^{\min}+2, \ldots, \tilde{v}^{\max}-1, \tilde{v}^{\max}\}, \tag{8.31}$$

where $\tilde{v}^{\min}$ *is given by*

$$\tilde{v}^{\min} = \max\left\{v^{\min}, vl^{\min} - (|S|-1)v^{\max}, \underline{I} - |B|(|S|-1)v^{\max}\right\}. \tag{8.32}$$

The upper bound $\tilde{v}^{\max}$ *is given by*

$$\tilde{v}^{\max} = \min\left\{m^{\max}v^{\max}, m^{\max}(vl^{\max} - (|S|-1)v^{\min}), \bar{I} - |B|(|S|-1)v^{\min}\right\}. \tag{8.33}$$

We restrict the set N of possible single supplies by regarding the bounds for supplied items from the formulation of the SLDP($W^{E'}$), Problem 7. We satisfy the minimum supply per branch and size by regarding the minimum number $v_{\min}$ of items per lot-type and size. The minimum number of items $vl^{\min}$ contained in a lot-type complies the minimum supply per branch. The maximum supply per branch and size is $v^{\max}$. We consider a particular size in a branch: The overall supply for the remaining $|S|-1$ sizes in this branch is bounded above by $(|S|-1)v^{\max}$. There are only positive numbers of supplied items possible. Such, $\tilde{v}^{\min}$ can be bounded below by $\tilde{v}^{\min} \geq vl^{\min} - (|S|-1)v^{\max}$. Analogously we derive a bound for $\tilde{v}^{\min}$ by regarding the maximum overall supply. If we consider a size and a branch, then the overall supply for the remaining sizes per branch is bounded below by $\underline{I} - |B|(|S|-1|)v^{\max}$.

The upper bound $\tilde{v}^{\max}$ can be derived in a similar way. It is $\tilde{v}^{\max} \leq m^{\max}v^{\max}$. In terms of lot-types the maximum supply per size and branch regarding the maximum multiplicity is bounded above by $m^{\max}(vl^{\max} - (|S|-1)v^{\min})$. In terms of the upper bound for the overall supply $\overline{I}$ the third bound can be derived.

In Problem 12 we restrict us to overall supplies per branch that lie in between bounds $\tilde{vl}^{\min}$ and $\tilde{vl}^{\max}$.

Observation 4 (minimum and maximum supply per branch)**.**
The lower bound $\tilde{vl}^{\min}$ in Problem 12 can be chosen as

$$\tilde{vl}^{\min} = \max\left\{vl^{\min}, \underline{I} - (|B|-1)m^{\max}vl^{\max}\right\}. \tag{8.34}$$

The upper bound $\tilde{vl}^{\max}$ can be chosen as

$$\tilde{vl}^{\max} = \min\left\{m^{\max}vl^{\max}, \overline{I} - (|B|-1)vl^{\min}\right\}. \tag{8.35}$$

With $vl^{\min}$ being the minimum supply per lot-type and 1 the minimum multiplicity in the non-relaxed formulation, the minimum supply per branch $\tilde{vl}^{\min}$ can be bounded below by $vl^{\min}$. Now we focus on a branch b. The maximum supply for the remaining $|B|-1$ branches is bounded above by $m^{\max}vl^{\max}$. This means, if we supply the remaining branches with the maximum possible number of items $m^{\max}vl^{\max}$ at least we have to deliver $\underline{I} - (|B|-1)m^{\max}vl^{\max}$ items to Branch b to adhere to the lower bound for the overall supply.

We proceed similarly for $\tilde{vl}^{\max}$.

A greedy approach

We solve Problem 12 by Algorithm 8. In the algorithm we do not regard fixed markdown costs, we add them later on. Before we outline the approach we will state some definitions and properties the algorithm draws on.

Definition 20 (expected single supply revenue)**.** For a subset $E' \subseteq E$ of scenarios with a corresponding set of maps $W^{E'}$ containing a map $e \to t^e$ for each particular scenario $e \in E'$ we define the *expected single supply revenue* $\bar{\bar{a}}^{W^{E'}}_{b,s,n}$ for Branch $b \in B$, Size $s \in S$ and Number $n \in N$ as

$$\bar{\bar{a}}^{W^{E'}}_{b,s,n} := \sum_{e \in E'} \text{Prob}(e)\bar{a}^{e \to t^e}_{b,s,n}. \tag{8.36}$$

The expected single supply revenue is concave.

Corollary 10 (concavity of the expected single supply revenue)**.** *For a given set of maps $W^{E'}$ where each scenario $e \in E'$ is mapped to a price trajectory t^e it is $\sum_{e \in E'} \mathrm{Prob}(e) a_{b,s,n}^{e \to t^e}$ concave in n for every branch $b \in B$ and size $s \in S$.*

Proof. Because linear combinations of concave functions with non-negative coefficients are also concave – and the expected value is one – the claim follows from Theorem 5. □

Definition 21 (current supply and additional revenue)**.** With $supply(b, s) \in N$ we assign an integer supply to Branch b and Size s in Algorithm 8. The *additional revenue* $ac(b, s)$ for supplying $supply(b, s) + 1$ items of Size s to Branch b is given by

$$ac(b, s) := \bar{\bar{a}}^{W^{E'}}_{b,s,supply(b,s)+1} - \bar{\bar{a}}^{W^{E'}}_{b,s,supply(b,s)}. \tag{8.37}$$

Algorithm 8 Single supply relaxation

Require: set B of branches, set S of sizes, minimum single supply $\tilde{v}^{\min}$, maximum single supply $\tilde{v}^{\max}$, minimum supply per branch $\tilde{vl}^{\min}$, maximum supply per branch $\tilde{vl}^{\max}$, expected single supply revenues $\bar{\bar{a}}^{W^{E'}}_{b,s,n}$ for all numbers $n \in N$, all branches $b \in B$ and all sizes $s \in S$
Ensure: optimal objective value $cb := z^*_{\text{SLDP-CB}(W^{E'})}$ of Problem 12

1: init $I = |B||S|\tilde{v}^{\min}$
   init $cb = \sum_{b \in B} \sum_{s \in S} \bar{\bar{a}}^{W^{E'}}_{b,s,\tilde{vl}^{\min}} - \sum_{e \in E'} \mathrm{Prob}(e)(\mu_f \tilde{\delta}^{t^e}_{k_{\max}-1} + |B|\mathrm{pcost} + \delta_1)$
2: **for all** $b \in B$ **do**
3: **for all** $s \in S$ **do**
4: init $I_{b,s} = \tilde{v}^{\min}$
5: **end for**
6: init $I_b = |S|\tilde{v}^{\min}$
7: init $I = |S|\tilde{v}^{\min}$
8: **while** $I_b < \tilde{vl}^{\min}$ **do**
9: $s' = \arg\max_{s \in S: I_{b,s} \le \tilde{v}^{\max}} ac(b, s)$
10: $I_{b,s'} = I_{b,s'} + 1$
11: $I_b = I_b + 1$
12: $I = I + 1$
13: **end while**
14: **end for**
15: **while** $I < \underline{I}$ **do**
16: $(b', s') = \arg\max_{(b,s) \in B \times S: I_{b,s} \le \tilde{v}^{\max}, I_b \le \tilde{vl}^{\max}} ac(b, s)$
17: $I_{b',s'} = I_{b',s'} + 1$
18: $I_{b'} = I_{b'} + 1$
19: $I = I + 1$
20: $cb = cb + ac(b', s')$
21: **end while**
22: **while** $I \le \bar{I}$ **do**
23: **if** $\max_{b \in B, s \in S: I_{b,s} \le \tilde{v}^{\max}, I_b \le \tilde{vl}^{\max}} ac(b, s) \le 0$ **then**
24: **break**
25: **end if**
26: $(b', s') = \arg\max_{(b,s) \in B \times S: I_{b,s} \le \tilde{v}^{\max}, I_b \le \tilde{vl}^{\max}} ac(b, s)$
27: $I_{b',s'} = I_{b',s'} + 1$
28: $I_{b'} = I_{b'} + 1$
29: $I = I + 1$
30: $cb = cb + ac(b', s')$
31: **end while**
32: **return** cb

At the beginning of Algorithm 8, in Step 4, we ensure that the bounds for minimum supply per size and per branch are adhered to. The next is the abidance of the minimum supply per branch. For each branch we determine the branch with the highest additional

revenue until we meet $vl^{\min}$, Step 8. We consider only branches and sizes for which increasing the supply would not exceed the upper bound $\tilde{v}^{\max}$. Ties are always broken arbitrarily.

After guaranteeing the minimum supply per branch, the focus is on the lower bound $\underline{I}$ for the overall supply. Now we choose the size s' and branch b' with highest additional revenue. The current supply $I_{b',s'}$ is increased by one. Again we consider only branches and sizes for which increasing the supply would not exceed the upper bounds $\tilde{v}^{\max}$ and $\tilde{vl}^{\max}$. This is done by traversing the while-loop in Step 15. After these steps we have a feasible solution for Problem 12. To obtain the optimum we greedily choose a branch and size with highest positive additional revenue. We increase the corresponding supply $I_{b',s'}$ by one. This is done in the while-loop in Step 22. If there is no branch and size with positive additional revenue, we are already optimal. Otherwise the last step is repeated as long as the maximum additional profit $ac(b,s)$ is positive or until we meet the upper bound $\bar{I}$ for the overall supply.

In the algorithm the concavity of the expected single supply revenues is exploited. Because we know that the additional revenue for one additional item does not increase for growing n, greedily choosing the size and branch which highest positive additional revenue yields the optimal solution of Problem 12.

Remark 6 (runtime of Algorithm 8)**.** *By storing the additional revenues $ac(b,s)$ in heaps the complexity of Algorithm 8 is given by*

$$\mathcal{O}(|B||S| + |B|(vl^{\min} - v^{\min})\log(|S|) + (\bar{I} - |B|vl^{\min})\log(|B||S|)). \tag{8.38}$$

8.2.2 Extended wait-and-wee solutions for ISPO

In the enumeration tree for ISPO, see Section 7.4, we fix a price trajectory t^e to every scenario $e \in E$ according to the set of maps W^E. To obtain the optimal first stage decision for a particular scenario e we solve an SLDP($W^{E'}$), Problem 7, with $E' = \{e\}$ and objective function

$$\max \text{Prob}(e) \left(\sum_{b\in B} \sum_{\ell\in L} \sum_{m\in M} \hat{a}^{e\to t^e}_{b,\ell,m} x_{b,\ell,m} - \mu_f \tilde{\delta}^{t^e}_{k_{\max}-1} - \sum_{i=1}^{\kappa} \delta_i \cdot z_i \right). \tag{8.39}$$

Notation 5 (extended wait-and-see solution for ISPO)**.** We denote the extended wait-and-see solution for ISPO with fixed price trajectories according to the set W^E of maps for $E' \subseteq E$ by $\tilde{WS}^{E'}(W^E)$.

Observation 5 (extended wait-and-see solution for ISPO)**.** *We consider a set of maps W^E where each scenario $e \in E$ is mapped to a price trajectory and a subset of scenarios $E' \subseteq E$. The optimal objective values for the* SLDP($\{e \to t^e\}$) *with $e \in E \setminus E'$ are denoted by $z^*_{\text{SLDP}(\{e\to t^e\})}$ and for the* SLDP($W^{E'}$) *by $z^*_{\text{SLDP}(W^{E'})}$.*

Then the extended wait-and-see solution $\tilde{WS}^{E'}(W^E)$ is given by

$$\tilde{WS}^{E'}(W^E) = z^*_{\text{SLDP}(W^{E'})} + \sum_{e\in E\setminus E'} z^*_{\text{SLDP}(\{e\to t^e\}}. \tag{8.40}$$

Because – as we will see in Chapter 9 – the SLDP($W^{E'}$)s for real-world instances are hard to solve, we only use relaxed (extended) wait-and-see solutions in our customized Branch&Bound solver. On the one side we use LP relaxations, i.e. we solve

the related SLDP-LP($W^{E'}$), on the other side single supply relaxations obtained by solving the SLDP-CB($W^{E'}$) via Algorithm 8.

Notation 6 (LP-relaxed extended wait-and-see solution – LPB, ELPB)**.** We denote the LP-relaxed extended wait-and-see solution or extended LP bound ELPB for ISPO with fixed price trajectories for each scenario according to the set $W^{E'}$ of maps for the set E' of scenarios with $E' \neq \emptyset$ by $WS_{lp}^{E'}(W^{E'})$. For $E' = \emptyset$ the LP-relaxed extended wait-and-see solution equals the LP-relaxed wait-and-see solution and is denoted by LPB.

Observation 6 (LP-relaxed extended wait-and-see solution – LPB, ELPB)**.** *We consider a set of maps W^E where each scenario $e \in E$ is mapped to a price trajectory and a subset of scenarios $E' \subseteq E$.*

Then the LP-relaxed extended wait-and-see solution $\tilde{\mathrm{WS}}_{lp}^{E'}(W^E)$ is given by

$$\tilde{WS}_{lp}^{E'}(W^E) = z^*_{\mathrm{SLDP-LP}(W^{E'})} + \sum_{e \in E \setminus E'} z^*_{\mathrm{SLDP-LP}(\{e \to t^e\})}. \tag{8.41}$$

Notation 7 (single-supply-relaxed extended wait-and-see solution, (extended) combinatorial bound – CB, ECB)**.** We denote the single-supply-relaxed extended wait-and-see solution or extended combinatorial bound ECB for ISPO with fixed price trajectories for the set of maps W^E for $E' \neq \emptyset$ by $WS_{\mathrm{cb}}^{E'}(W^E)$. For $E' = \emptyset$ the single-supply-relaxed extended wait-and-see solution equals the single-supply-relaxed classical wait-and-see solution and is denoted by CB.

Observation 7 (single-supply-relaxed extended wait-and-see solution, (extended) combinatorial bound – CB, ECB)**.** *We consider a set of maps W^E where each scenario $e \in E$ is mapped to a price trajectory and a subset of scenarios $E' \subseteq E$.*

Then the single-supply-relaxed extended wait-and-see solution $\tilde{WS}_{cb}^{E'}(W^E)$ for a subset $E' \subseteq E$ of scenarios and the set of W^E of maps is given by

$$\tilde{WS}_{cb}^{E'}(W^E) = z^*_{\mathrm{SLDP-CB}(W^{E'})} + \sum_{e \in E \setminus E'} z^*_{\mathrm{SLDP-CB}(\{e \to t^e\})}. \tag{8.42}$$

We can mix up combinatorial and LP-relaxations for the computation of our relaxed (extended) wait-and-see solution. Both, the optimal objective value of the LP relaxation SLDP-LP($W^{E'}$) and the optimal objective value of the SLDP-CB($W^{E'}$), yield dual bounds for the SLDP($W^{E'}$). For all subsets of scenarios E_1 and E_2 with $E_1 \subseteq E \setminus E'$, $E_2 \subseteq E \setminus E'$ and $E_1 \dot{\cup} E_2 = E \setminus E'$ the relaxed extended wait-and-see solutions

$$z^*_{\text{SLDP-CB}(W^{E'})} + \sum_{e \in E_1} z^*_{\text{SLDP-CB}(\{e \to t^e\})} + \sum_{e \in E_2} z^*_{\text{SLDP-LP}(\{e \to t^e\})} \tag{8.43}$$

just like

$$z^*_{\text{SLDP-LP}(W^{E'})} + \sum_{e \in E_1} z^*_{\text{SLDP-CB}(\{e \to t^e\})} + \sum_{e \in E_2} z^*_{\text{SLDP-LP}(\{e \to t^e\})}. \tag{8.44}$$

are dual bounds for the SLDP(W^E).

In our exact Branch&Bound solver, see Chapter 9, we will mix up combinatorial and LP relaxations. Because – as we will see in the next section – the computation of the (extended) combinatorial bounds is much faster than solving the LP relaxations of the SLDP($W^{E'}$), we will solve LP relaxations only on demand if combinatorial bounds are not tight enough to prune the tree.

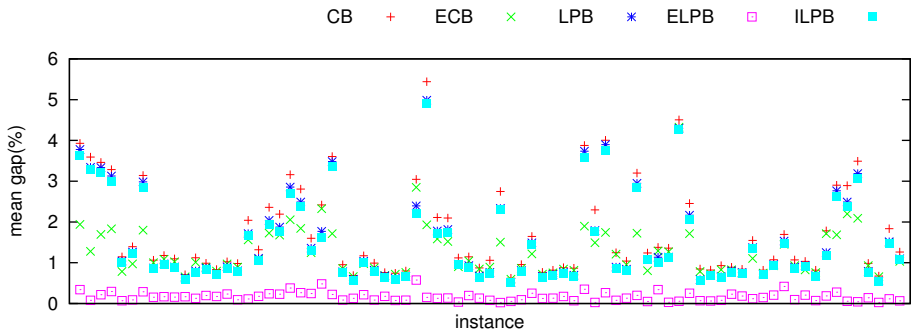

Figure 8.1: Goodness of dual bounds

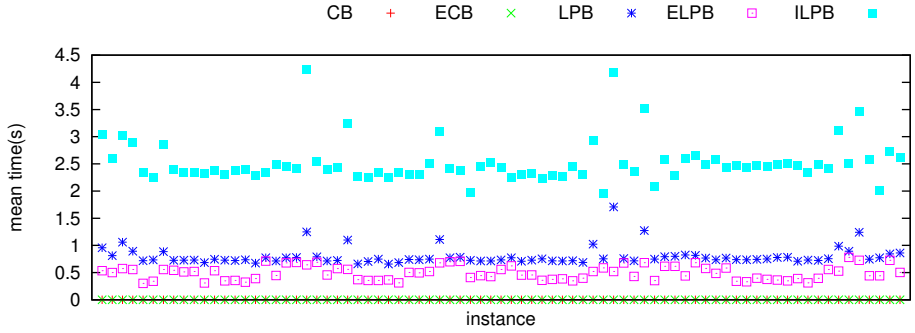

Figure 8.2: Computation time of dual bounds

8.2.3 Computational results

We compared the proposed bounds on the first 79 instances of our test set $\mathcal{I}_6$, see Appendix E.

We computed the bounds for all leaves of the enumeration tree of ISPO. In Figure 8.1 and Figure 8.2 we depicted results in terms of optimality gaps and computation time.

We denoted the bound resulting from the standard wait-and-see solution of the related SLDP(W^E) by "ILPB". For the other names see the previous notations.

For the extended bounds ECB and ELPB we always pooled all three scenarios. That means ELPB related to W^E in our case is the same as the optimal objective value of the SLDP-LP(W^E).

Averaged over all instances and rounded to two decimal figures we get optimality gaps of 1.74% for CB, 1.26% for ECB, 1.53% for LPB, 0.16% for ELPB and 1.47% for ILPB. The mean runtime is 0.00 seconds for CB and ECB, 0.80 seconds for LPB, 0.49 seconds for ELPB and 2.53 seconds for ILPB. Please note that for an LPB the numbers of the SLDP-LP($W^{E'}$)s that have to be solved amounts to $|E| = 3$, for an ELPB just one, namely SLDP-LP(W^E).

The wait-and-see solutions based on single-supply-relaxed bounds, as well the combinatorial bounds CB as the extended combinatorial bounds ECB, are as a rule weaker than the wait-and-see solutions LPB and ELPB based on LP relaxations, but can be solved much faster. ECB and ELPB in many cases also beat the standard wait-

and-see solution ILPB. For ELPB this is even always the case. As already mentioned ELPB for $E' = E$ is the same as the LP relaxation SLDP-LP(W^E) of the SLDP(W^E). This means that the SLDP(W^E) has a small integrality gap, i.e. the optimal objective value of SLDP-LP(W^E) is not far from the optimal objective value of the SLDP(W^E).

8.3 Conclusion of the chapter

We introduced new bounds for two-stage stochastic programs based on the wait-and-see solution from stochastic programming. The extended wait-and-see solutions are tighter than the classical wait-and-see solution. We applied these bounds on the leaves of the enumeration tree of ISPO – here each leaf corresponds to an SLDP(W^E). Because the underlying binary programs are hard to solve we relax them – either by allowing single supply instead of lot-types or by the LP relaxation. The results for a set of small test instances show that the bounds based on the single supply relaxation can be obtained very fast. The relaxed extended wait-and-see solutions are always faster to compute than the classical wait-and-see solution. In some cases they even beat the classical wait-and-see solution in terms of the optimality gap; for the LP-relaxed extended wait-and-see solution this is even always the case.

Chapter 9

Solving the Integrated Size and Price Optimization Problem

Now we present two of the main results of this thesis: our solvers for the Integrated Size and Price Optimization problem ISPO. Since the MIP formulation of ISPO presented in Chapter 6 for real instances cannot be solved directly by state-of-the-art MIP solvers, we developed two approaches: An exact algorithm for benchmarking and a fast heuristic for practical use. In the exact algorithm we exploit the fact that for fixed price trajectories ISPO reduces to an SLDP, Chapter 7. Dual bounds on the base of the wait-and-see solution, see Chapter 8, allow us to prune the enumeration tree from Section 7.4. We outline the resulting Branch&Bound algorithm named ISPO-BAB in Section 9.1. The heuristic solver, ISPO-PingPong, is presented in Section 9.2. It exploits the fact that for every valid second stage decision, i.e. for every set of maps "scenario¸to price trajectory" for all scenarios there exists a valid first stage decision, a valid supply policy. We call this property *reversible recourse*. We present computational results for both solvers on real-world instances in Section 9.3 and give some remarks about the goodness of our proposed heuristic in Section 9.4, before we conclude this chapter in Section 9.5.

9.1 An exact Branch&Bound approach

Now we present our exact solver for ISPO – a customized Branch&Bound algorithm. In Chapter 7, Section 7.4, we already showed how to solve ISPO by enumerating all possible combinations of assignments of price trajectories to scenarios. We adopt this idea and develop it further by applying the dual bounds presented in Chapter 8. The result is our exact Branch&Bound algorithm ISPO-BAB. A node at Depth j corresponds to all maps "scenario to price trajectory" with the images of the first j scenarios fixed. The leaves are the maps with fixed images for all scenarios. In the branching step we extend a partially defined set of maps at a node by maps to all valid price trajectories for the next scenario.

9.1.1 The algorithm

We present the detailed implementation of the above concept – Algorithm 9.

In a first step we compute for each map "scenario to price trajectory" the single supply revenues for each branch and size and every number n from the set N of integer supplies, see Theorems 3 and 4, by Step 4. We apply a simple dominance check, Step 6, in Algorithm 10: If for each number, each size and each branch for one price trajectory t the single supply revenue is always smaller than the single supply revenue for another price trajectory t' in terms of the same scenario, the price trajectory t is dominated by t'. If this is the case we can exclude t from further consideration for this scenario. The set of non-dominated price trajectories for scenario e is denoted by T^e.

Then, in Step 10 of Algorithm 9, we solve for each scenario $e \in E$ and each price trajectory $t \in T^e$ the SLDP-CB($\{e \to t\}$), Problem 12, and store the optimal objective function value – it is a summand for the computation of the relaxed wait-and-see solution – as $\Gamma(e \to t)$. We update these addends possibly later on by solving the LP-relaxation SLDP-LP($\{e \to t\}$). In further progress we add up these values to obtain relaxed wait-and-see solutions for the particular SLDP(W^E)s. But from these combinatorial bounds we also benefit in another way: These bounds are used to get an idea how good a price trajectory fits to a scenario. By ordering the trajectories decreasingly according to their combinatorial bounds in the Branch&Bound tree we expect to find the optimal solution for ISPO as early as possible which may lead to the possibility to prune a bigger part of the tree.

For the purpose of computing dual bounds also at the inner nodes of the tree we save the maximum lower bound for each scenario as $\Gamma^{\max}(e)$ in Step 13 and in the next step the index of the corresponding price trajectory as $\arg^{\max}(e)$.

Then we start a depth-first-search. We start with an empty set W^{E_0} of maps "scenario→price trajectory", Step 18, and set the current depth to value one, Step 19.

The depth-first-search itself is outlined as Algorithm 11: We fix the first price trajectory in the order to the scenario related to the current depth, Step 1, and compute the dual bound in terms of a relaxed wait-and-see solution, Step 2. If we are in the first branch of the tree, we abandon to compute improved lower bounds because we will not be able to prune the tree until the primal bound is updated for the first time. Otherwise – if the dual bound is to weak to prune the tree – we try to improve the current bound by updating the bounds $\Gamma(e \to t)$ by LP-relaxations or by computing extended wait-and-see solutions in Step 6, more precisely by the algorithms 12,13 and 14. If the dual bound is smaller than the current primal bound, we are able to prune the current branch at this point.

Whenever we reach a leaf of the tree at depth $|E|$ and are not able to prune, we solve the corresponding SLDP(W^E) and possibly update the primal bound.

In Algorithm 12 the summands for the computation of the LP-relaxed wait-and-see solution are computed. We avoid to solve more LP relaxations of the SLDP($W^{E'}$) than necessary. That means we mix combinatorial and LP-relaxed bounds.

We compute LP relaxations of the SLDP($W^{E'}$) just until the resulting "mixed" relaxed wait-and-see solution is smaller than the current primal bound. We also update the corresponding values $\Gamma(e \to t)$ if the SLDP-LP($\{e \to t\}$) yields a smaller optimal objective function value than the SLDP-CB($\{e \to t\}$), i.e. if the LP-relaxation of the SLDP($\{e \to t\}$) has a smaller optimality gap than the related single supply relaxation, Step 6. If this is the case, we may have to adapt the maximum bound for the current scenario $\Gamma(e)^{\max}$ in Step 7.

Algorithm 9 ISPO-BAB

Require: complete data of an instance of ISPO, set of price trajectories T
Ensure: expected revenue maximizing supply x^* in terms of lots and related price trajectories
1: init global primal bound $z^* = -\infty$
2: **for all** $e \in E$ **do**
3: **for all** $t \in T$ **do**
4: compute $\bar{a}_{b,s,n}^{e \to t} \forall b \in B, s \in S, n \in N$, Algorithm 6
5: **end for**
6: $T^e =$ ISPO-DOM(e)
7: **end for**
8: **for all** $e \in E$ **do**
9: **for all** $t \in T^e$ **do**
10: compute the optimal solution of SLDP-CB($\{e \to t\}$) and save the related optimal objective function value as $\Gamma\,(e \to t)$
11: **end for**
12: sort the price trajectories T^e in descending order according to $\Gamma\,(e \to t) \to$ set of indexed price trajectories $T^e = (t_1^e, t_2^e, \ldots, t_{|T|}^e)$
13: set $\Gamma^{\max}(e) = \Gamma\,(e \to t_1^e)$
14: set $\arg^{\max}(e) = t_1^e$
15: **end for**
16: sort the scenarios $\to$ set of indexed scenarios $E = \{e_1, \ldots, e_{|E|}\}$
17: **for all** $i = 1, \ldots, |T^{e_1}|$ **do**
18: set $W^{E_0} = \emptyset$
19: set $j = 1$
20: ISPO-DFS$(1,i,W^{E_0})$
21: **end for**
22: **return** z^* and related optimal solution of ISPO

9.1.2 Some implementational aspects

At this point we outline some implementational details of our Algorithm 9.

MIP solvers

As MIP solver or LP solver for the SLDP($W^{E'}$)s and the SLDP-LP($W^{E'}$)s one can choose between ILOG CPLEX and SCIP. By default we use CPLEX because it leads to shorter computation times. An advantage of CPLEX is that if once an SLDP-LP($W^{E'}$) is generated we can keep it in memory and only have to update the objective coefficients the next time we have to solve an SLDP-LP($W^{E'}$). Moreover we use the possibility to perform so-called warm starts in the simplex algorithm.

Warm starts

The constraints of the SLDP-LP($W^{E'}$) and SLDP($W^{E'}$) are independent from the fixed price trajectories, only the coefficients of the objective differ. So every optimal basis of an SLDP-LP($W^{E'}$) is also feasible for the SLDP-LP($W^{\tilde{E}}$)s with $W^{\tilde{E}} \neq W^{E'}$ at the other nodes of the tree.

When solving an SLDP-LP($W^{E'}$) we let CPLEX start the primal simplex with the optimal basis of an already solved SLDP-LP($W^{E'}$) – if available. For detailed information about the simplex algorithm we refer the reader to [Van07].

Additional Bounding

As soon as a primal bound for ISPO is found, we hand over this primal bound to the MIP solver. The MIP-solver uses this bound in its internal Branch&Bound approach to prune branches which can not lead to a better solution.

Algorithm 10 ISPO-DOM

Require: single supply revenues $\bar{a}^{e \to t}_{b,s,n}$ $\forall b \in B, s \in S, n \in N, \forall t \in T$ for the considered scenario e
Ensure: set of non-dominated ordered price trajectories T^e

```
1: init T^e = ∅
2: init dom_t = true
3: for all i = 1, ..., |T| do
4:     set t = t_i
5:     init tdomt' = true
6:     init t'domt = true
7:     if dom_t = true then
8:         continue
9:     end if
10:    for all j = i + 1, ..., |T| do
11:        set t' = t_j
12:        if dom_t' = true then
13:            continue
14:        end if
15:        for all b ∈ B do
16:            for all s ∈ S do
17:                for all n ∈ N do
18:                    if ā^{e_i→t}_{b,s,n} > ā^{e_i→t'}_{b,s,n} then
19:                        t'domt = false
20:                    else
21:                        if ā^{e_i→t}_{b,s,n} < ā^{e_i→t'}_{b,s,n} then
22:                            tdomt' = false
23:                        end if
24:                    end if
25:                    if tdomt' = t'domt = false then
26:                        goto 39
27:                    end if
28:                end for
29:            end for
30:        end for
31:        if tdomt' then
32:            set dom_t' = true
33:        else
34:            if t'domt then
35:                set dom_t = true
36:            end if
37:        end if
38:    end for
39:    if dom_t = false then
40:        set T^e = T^e ∪ {t}
41:    end if
42: end for
```

Algorithm 11 ISPO-DFS

Require: depth/scenario index j
width/index i of ordered price trajectories
bounds $\Gamma\left(e \to t^e\right)$ for all $e \in E$, $t^e \in T^e$
set of maps $W^{E_{j-1}}$ for $E_{j-1} := \{e_i | i < j\}$
current global upper bound z^*
Ensure: possibly updated bounds $\Gamma\left(e \to t^e\right)$ and/or updated primal bound z^*
$W^{E_j} = W^{E_{j-1}} \cup \{e_j \to t_i^{e_j}\}$
compute the (mixed) relaxed wait-and-see solution $\hat{\Gamma}\left(W^{E_j}\right)$ by
$\hat{\Gamma}\left(W^{E_j}\right) = \sum_{e \to t \in W^{E_j}} \Gamma\left(e \to t\right) + \sum_{o=j+1}^{|E|} \Gamma\left(e_o\right)^{\max}$
if $\hat{\Gamma}\left(W^{E_j}\right) \leq z^*$ **then**
  return
end if
try to improve the dual bound by
$\hat{\Gamma}\left(W^{E_j}\right) =$ISPO-ECB$\left(W^{E_j}, \hat{\Gamma}\left(W^{E_j}\right), j\right)$ and or
$\hat{\Gamma}\left(W^{E_j}\right) =$ISPO-ELPB$\left(W^{E_j}, \hat{\Gamma}\left(W^{E_j}\right), j\right)$ and or
$\hat{\Gamma}\left(W^{E_j}\right) =$ISPO-LPB$\left(W^{E_j}, \hat{\Gamma}\left(W^{E_j}\right), \left\{\Gamma\left(e \to t\right), e \to t \in W^{E_j}\right\}\right)$
if $\hat{\Gamma}\left(W^{E_j}\right) \leq z^*$ **then**
  return
end if
if $j < |E|$ **then**
  for all $i = 1, \ldots, |T^{e^{j+1}}|$ **do**
    ISPO-DFS$\left(j+1, i, W^{E_j}\right)$
  end for
else
  solve the SLDP(W^{E_j}) $\to$ optimal objective value $z^*_{\text{SLDP}\left(W^{E_j}\right)}$ and optimal supply policy $x^*_{\text{SLDP}\left(W^{E_j}\right)}$
  if $z^*_{\text{SLDP}\left(W^{E_j}\right)} > z^*$ **then**
    $z^* = z^*_{\text{SLDP}\left(W^{E_j}\right)}$
    $x^* = x^*_{\text{SLDP}\left(W^{E_j}\right)}$
  end if
end if

Algorithm 12 ISPO-LPB

Require: set of maps W^{E_j} of price trajectories fixed to scenarios
dual bound $\hat{\Gamma}\left(W^{E_j}\right)$ for the given set of maps
dual bounds $\Gamma\left(e \to t\right)$ for each single map $e \to t$ in W^{E_j}
Ensure: possibly improved dual bound $\hat{\Gamma}\left(W^{E_j}\right)$ based on the LP-relaxed wait-and-see solution
possibly improved dual bounds $\Gamma\left(e_o \to t^{e_o}\right)$ for $o \leq j$
for all $o = 1, \ldots, j$ **do**
  if SLDP-LP($\{e_o \to t^{e_o}\}$) not solved until now **then**
    solve SLDP-LP($\{e_o \to t^{e_o}\}$) $\to$ optimal objective function value $z^*_{\text{SLDP-LP}(\{e_o \to t^{e_o}\})}$
    if $z^*_{\text{SLDP-LP}(\{e_o \to t^{e_o}\})} < \Gamma\left(e_o \to t^{e_o}\right)$ **then**
      set $\Gamma_{old} = \Gamma\left(e_o \to t^{e_o}\right)$
      $\Gamma\left(e_o \to t^{e_o}\right) = z^*_{\text{SLDP-LP}(\{e_o \to t^{e_o}\})}$
      if $\arg^{\max}(e_o) = t^{e_o}$ **then**
        $\arg(e_o)^{\max} = \arg\max_{t' \in T^{e_o}} \{\Gamma\left(e_o \to t'\right)\}$
        $\Gamma^{\max}(e_o) = \max_{t' \in T^{e_o}} \{\Gamma\left(e_o \to t'\right)\}$
      end if
      $\hat{\Gamma}\left(W^{E_j}\right) = \hat{\Gamma}\left(W^{E_j}\right) - \Gamma_{old} + z^*_{\text{SLDP-LP}(\{e_o \to t^{e_o}\})}$
      if $\hat{\Gamma}\left(W^{E_j}\right) < z^*$ **then**
        return
      end if
    end if
  end if
end for
return $\hat{\Gamma}\left(W^{E_j}\right)$

Algorithm 13 ISPO-ECB

Require: set of maps W^{E_j} of price trajectories fixed to scenarios
dual bound $\hat{\Gamma}\left(W^{E_j}\right)$ for the given set of maps
current depth/scenario index j
Ensure: possibly improved dual bound $\hat{\Gamma}\left(W^{E_j}\right)$ based on the single-supply-relaxed extended wait-and-see solution
1: solve SLDP-CB(W^{E_j})
2: **if** $z^*_{\text{SLDP-CB}\left(W^{E_j}\right)} + \sum_{o=j+1}^{|E|} \Gamma\left(e_o\right)^{\max} < \hat{\Gamma}\left(W^{E_j}\right)$ **then**
3: $\quad \hat{\Gamma}\left(W^{E_j}\right) = z^*_{\text{SLDP-CB}\left(W^{E_j}\right)} + \sum_{o=j+1}^{|E|} \Gamma(e_o)^{\max}$
4: **end if**
5: **return** $\hat{\Gamma}\left(W^{E_j}\right)$

Algorithm 14 ISPO-ELPB

Require: set of maps W^{E_j} of price trajectories fixed to scenarios
dual bound $\hat{\Gamma}\left(W^{E_j}\right)$ for the given set of maps
current depth/scenario index j
Ensure: possibly improved dual bound $\hat{\Gamma}\left(W^{E_j}\right)$ based on the LP-relaxed extended wait-and-see solution
1: solve SLDP-LP $\left(W^{E_j}\right)$
2: **if** $z^*_{\text{SLDP-LP}\left(W^{E_j}\right)} + \sum_{o=j+1}^{|E|} \Gamma\left(e_o\right)^{\max} < \hat{\Gamma}\left(W^{E_j}\right)$ **then**
3: $\quad \hat{\Gamma}\left(W^{E_j}\right) = z^*_{\text{SLDP-LP}\left(W^{E_j}\right)} + \sum_{o=j+1}^{|E|} \Gamma^{\max}(e_o)$
4: **end if**
5: **return** $\hat{\Gamma}\left(W^{E_j}\right)$

9.1.3 Computational results

We compare different settings for ISPO-BAB in terms of the applied dual bounds. We performed ISPO-BAB for all 166 instances from our test-set $\mathcal{I}_6^{\text{test}}$, see Appendix E. For these tests we first abandon dominance checks, Algorithm 10. We compare different combinations of combinatorial bounds CB, extended combinatorial bounds ECB, LP bounds LPB and extended LP bounds ELPB. Additionally we applied dominance "DOM" in terms of the price trajectories to reduce the size of our Branch&Bound tree a priori. In Figure 9.1 we see how many leaves of the tree can not be pruned by applying the given combinations of bounds.

In many cases the number of remaining leaves can be much reduced by applying the extended combinatorial bounds against the case where only combinatorial bounds are used (from averagely 1.96% to 1.32%). If we use combinatorial bounds together with LP bounds or extended LP bounds there are averagely only 0.89% or 0.80% leaves remaining. Combining combinatorial bounds, extended combinatorial bounds and LP bounds together yields averagely 0,68% remaining leaves. If we replace in this case the LP bounds by extended LP bounds we can reduce the number of remaining leaves averagely to 0.65%.

We depict the improvements by additionally applying dominance in terms of using combinatorial, extended combinatorial and extended LP bounds in Figure 9.2. We see how strong we can reduce the number of remaining leaves in the tree "not pruned by dom." by applying dominance checks for the price trajectories a priori – averagely to 1.68 percents. While the number of remaining leaves in the Branch&Bound tree – or to be solved SLDP(W^E)s – only reduces about averagely 0.18 percents "diff ILPs" and

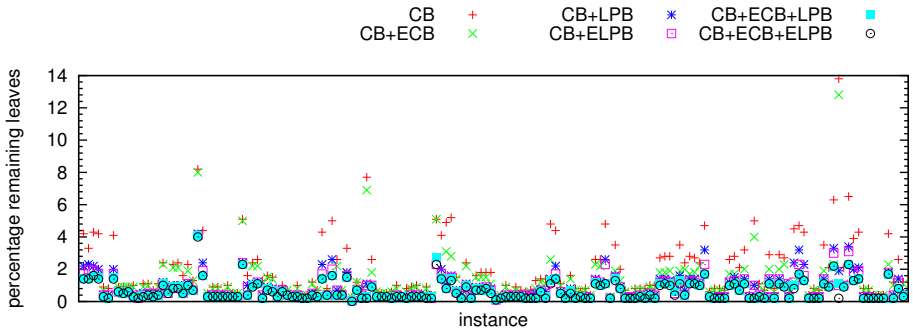

Figure 9.1: ISPO-BAB – percentage of non-pruned leaves applying different bounds

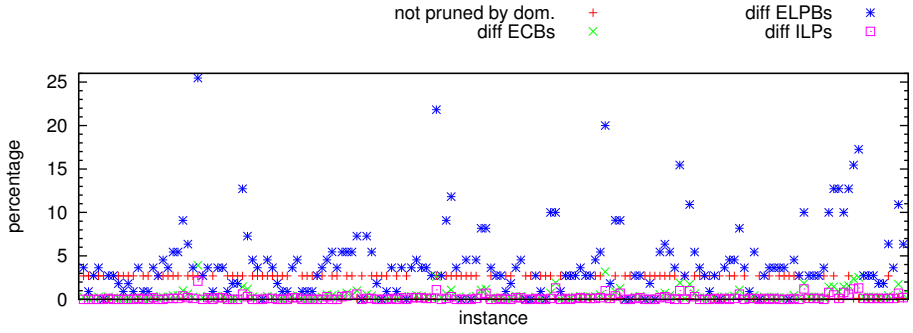

Figure 9.2: ISPO-BAB – Applying dominance for the price trajectories

the number of to be solved extended combinatorial bounds “diff ECBs” to 0.44 percents we get averagely about 4.33 percents extended LP bounds “diff ELPBs” fewer to solve by applying dominance checks for price trajectories.

Now we want to concentrate our attention on the computation time of ISPO-BAB in terms of using different dual bounds. For every instance from the set $\mathcal{I}_6^{\text{test}}$ for the stated bounds the average computation time at the leaves is depicted in Figure 9.3. If we neglect the scaling Figure 9.3 looks like Figure 9.1. Roughly speaking the figures illustrate a typical property of Branch&Bound algorithms: the worse the dual bounds the higher the computation times. In average we get 54,31 seconds for just using combinatorial bounds, 40.33 seconds for additionally using extended combinatorial bounds, 38.21 seconds for combinatorial and LP bounds, 38.36 for the combination of combinatorial bounds and extended LP bounds, 32.83 seconds by combining combinatorial, extended combinatorial and LP bounds and 34.19 seconds for using combinatorial, extended combinatorial and extended LP bounds.

Averagely 0.03% fewer ILPs have to be solved by applying ELPBs instead of LPBs.

If we add dominance in the last case we can reduce the runtime averagely to 21.73 seconds.

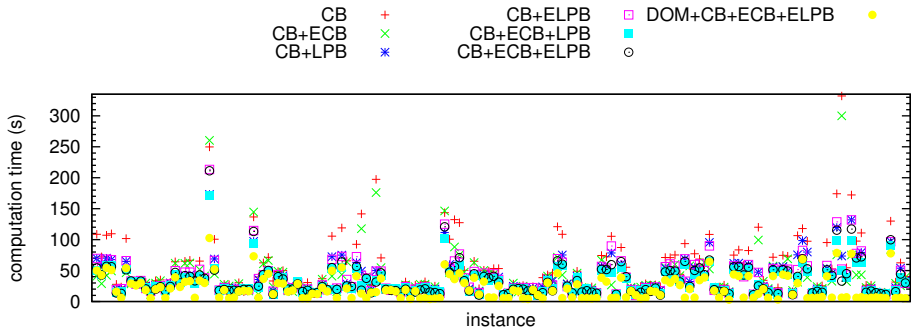

Figure 9.3: ISPO-BAB – solving time applying different bounds

9.1.4 ISPO-BAB applied to the accompanying example

We apply Algorithm 9 to our accompanying example from Chapter 4.

Example 10. A first step in the algorithm is the computation of the optimal objective function value of the SLDP-CB($\{e \rightarrow t\}$) for each price trajectory t and scenario e. Therefore we use the single supply revenues, Algorithm 6. The results for each scenario, branch and size can be found in Appendix C. Based on these data the single supply relaxation SLDP-CB($\{e \rightarrow t\}$) is solved to optimality via Algorithm 6. The results are stated in the following tables. For each price trajectory we state its index and the optimal supply for each pair (b,s) with $b \in B$ and $s \in S$. The optimal objective function values of the SLDP-CB($\{e \rightarrow t\}$) are always stated in the last column.

		low seller				
price trajectory	index	(1,S)	(1,L)	(2,S)	(2,L)	bound
$(0,0,0,0,3)$	0	3	3	2	2	18.86
$(0,0,0,1,3)$	1	3	3	2	2	18.31
$(0,0,0,2,3)$	2	3	3	2	2	17.64
$(0,0,1,1,3)$	3	3	4	1	2	16.86
$(0,0,1,2,3)$	4	3	4	1	2	15.96
$(0,0,2,2,3)$	5	3	4	1	2	14.39

		normal seller				
price trajectory	index	(1,S)	(1,L)	(2,S)	(2,L)	bound
$(0,0,0,0,3)$	0	2	4	1	3	51.58
$(0,0,0,1,3)$	1	2	4	1	3	51.10
$(0,0,0,2,3)$	2	2	4	1	3	51.10
$(0,0,1,1,3)$	3	2	4	1	3	51.09
$(0,0,1,2,3)$	4	2	4	1	3	50.61
$(0,0,2,2,3)$	5	2	4	1	3	51.09

		high seller				
price trajectory	index	(1,S)	(1,L)	(2,S)	(2,L)	bound
$(0,0,0,0,3)$	0	2	4	1	3	31.00
$(0,0,0,1,3)$	1	2	4	1	3	30.71
$(0,0,0,2,3)$	2	2	4	1	3	30.71
$(0,0,1,1,3)$	3	2	4	1	3	30.71
$(0,0,1,2,3)$	4	2	4	1	3	30.41
$(0,0,2,2,3)$	5	2	4	1	3	30.71

We order the price trajectories according to the revenues for the particular scenario and get:

scenario	1st	2nd	3rd	4th	5th	6th
low	0	1	2	3	4	5
normal	0	1	2	3	5	4
high	0	1	2	3	5	4

To illustrate our Branch&Bound approach we first abandon excluding price trajectories by dominance.

For this example we order the scenarios in the sequence "high-normal-low" and start with the depth-first-search, Algorithm 11.

The related Branch&Bound tree is depicted in Figure 9.4. The combinatorial bounds CB for the visited nodes are stated. The indices of the mapped price trajectories are always stated in the nodes. In the depth-first-search the nodes are visited in the order according to the numbers right above the nodes.

At first we fix the first price trajectory for the "high seller" scenario. For this scenario we obtain an expected revenue of 31.00. For the remaining scenarios yet we have not fixed a price trajectory: In terms of bounding we have always to consider the price trajectory that yields the highest revenue in terms of a single supply. For both, the "normal" and the "low" scenario, this is the first trajectory which yields 51.58 and 18.86 of revenue. Thus, the combinatorial bound amounts to $31.00 + 51.58 + 18.86 + 101.44$. The bound stays the same for the next two nodes. At Depth 3 we solve the SLDP(W^E) for $W^E = \{\text{high seller} \to 0, \text{normal seller} \to 0, \text{low seller} \to 0\}$. This yields a primal bound of 101.38.

We continue with fixing the first price trajectory for the low seller scenario. This yields an upper bound of 100.88. Because it is $100.88 < 101.44$ we prune the current branch. This is also the same for fixing the remaining price trajectories to the low seller scenario at this point.

We continue with fixing Price Trajectory 1 to the normal seller scenario. The related objective function value of the SLDP-CB amounts to 51.10. For the non-set low seller scenario again we have to add the expected revenue of 18.86. The expected revenue for the high seller scenario stays 31.00 – because we still fix Price Trajectory 0 to it. Thus, for the set of maps $W^{E'} = \{\text{high seller} \to 0, \text{normal seller} \to 1\}$ our current dual bound is 100.96. Because $100.96 < 101.38$ we prune the current branch. This is also the same for the following nodes at Depth 2.

Now we fix Price Trajectory 1 to the high seller scenario. This results in an expected revenue of 30.71. Again, we have to choose the maximum combinatorial bound for the remaining scenarios to obtain an upper bound for the wait-and-see solutions of the related childnodes. Altogether this yields a revenue of 101.14 and $101.14 < 101.38$. We can prune the tree also for the remaining nodes at Depth 1 and at the end obtain the optimal solution value of 101.44. In the related optimal solution we just delivered Lot-type (1,1). Both branches obtain this lot-type in Multiplicity 3.

In this example the computation of other bounds than the combinatorial bound CB was not necessary because we were able to prune all branches of the tree except the first one directly.

We did not apply dominance rules for the price trajectories. If we regarded dominance, the price trajectory with index 0 would dominate all other price trajectories for every particular scenario. Only the first branch of the Branch&Bound tree would remain.

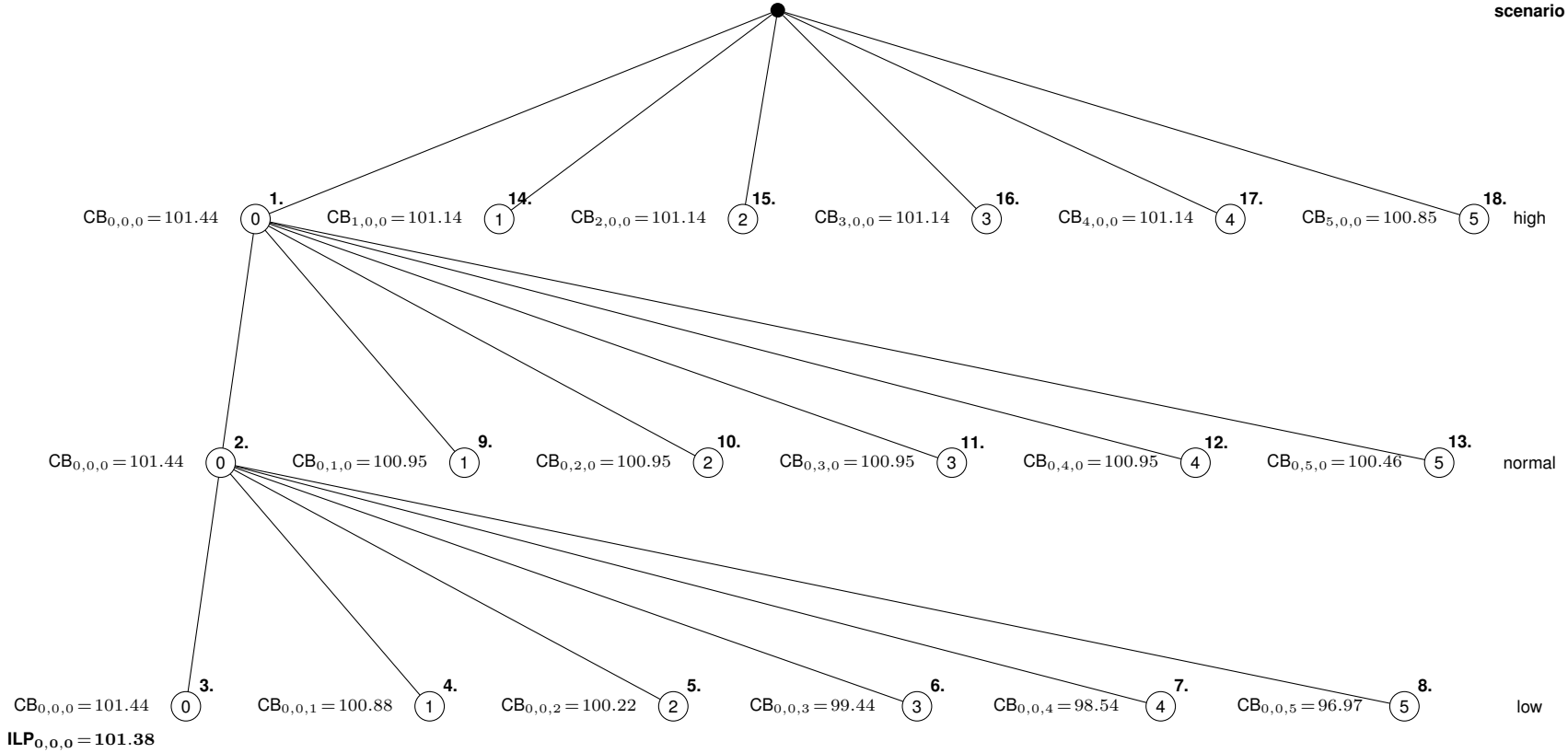

Figure 9.4: Example for ISPO-BAB

9.2 A heuristic approach – ISPO-PingPong

Our exact solver ISPO-BAB, Algorithm 9, which we presented in the preceding section is – as we will see in the next section – still too slow to satisfy real-world requirements. For practical purposes we therefore developed the heuristic ISPO-PingPong. The basic idea of this approach is to perform size optimization and price optimization alternately until convergence. There is a crucial property of ISPO that makes this approach possible. We call it *reversible recourse*. It is topic of the first subsection.

The main method of our heuristic ISPO-PingPong is presented in Subsection 9.2.2. In the following three subsections we will go more into detail. In Subsection 9.2.3 we describe the initial step of the heuristic – the determination of a set of maps "scenario to price trajectory". We present the approach to handle the size optimization stage in Subsection 9.2.4. In Subsection 9.2.5 we go into the other part of the algorithm – the price optimization stage. We present computational results for our heuristic in Subsection 9.2.6. Concluding, in Subsection 9.2.7 we will bring ISPO-PingPong into line with familiar approaches from literature.

9.2.1 Reversible recourse

Our problem has a special structure. We call it *reversible recourse*. This means: the *independent second stage* variables (in our case the price trajectories assigned to the different scenarios) can be interpreted as *independent first stage* decisions. The independent first stage variables and all dependent variables can then be seen as second stage variables. In our setting, fixing the independent decision variables of one stage does not even imply any restrictions to the feasible set of the independent variables in the other stage. Fixing a price trajectory to a scenario does not influence the feasibility of supply. This property can easily be observed in our Branch&Bound tree from the last Section, more precisely at the leaves which are related to an SLDP(W^E). The constraints in the SLDP(W^E) are for every leaf the same. Only the coefficients in the objective function differ.

9.2.2 The main algorithm

We outline our heuristic ISPO-PingPong in a top-down design. We describe the framework in Algorithm 15. After fixing a price trajectory to each scenario we alternate size and price optimization until the related objective function value of ISPO no longer increases. The particular parts are treated in the subsequent subsections.

9.2.3 Fixing price trajectories

At the beginning of Algorithm 15 we fix a price trajectory to each particular scenario. There are different possibilities to do this. In order to start with a supposed good assignment for each scenario e as a rule we solve the SLDP-CB($\{e \to t\}$) for all valid price trajectories t to optimality by Algorithm 8 and pick the price trajectory with the highest optimal objective function value.

We will present results for different settings in terms of fixing price trajectories in Subsection 9.2.6.

Algorithm 15 ISPO-PingPong

Require: complete data of an instance of ISPO
Ensure: optimal supply policy x^*
1: **for all** scenarios $e \in E$ **do**
2: fix a price trajectory for each scenario $\rightarrow$ set of maps W^E
3: **end for**
4: set $z^* = -\infty$
5: **while true do**
6: ISPO-SF(z^*): solve SLDP(W^E) exactly or heuristically $\rightarrow$ resulting supply x with expected revenue $z_{\text{SLDP}(W^E)}$
7: **if** $z_{\text{SLDP}(W^E)} = z^*$ **then**
8: **break**
9: **end if**
10: $x^* = x$
11: $z^* = z_{\text{SLDP}(W^E)}$
12: ISPO-PO(x^*): for all $e \in E$ solve POP^e for an initial stock according to x^* to optimality $\rightarrow$ updated set of maps W^E with expected revenue $z^*_{\text{POP}(x^*)}$
13: **if** $z^*_{\text{POP}(x^*)} = z^*$ **then**
14: **break**
15: **end if**
16: $z^* = z^*_{\text{POP}(x^*)}$
17: **end while**

9.2.4 Solving the SLDP(W^E)

To solve the SLDP(W^E) within ISPO-PingPong we implemented an adapted version of the SFA heuristic from Chapter 2.

We call this adapted version ISPO-SF, see Algorithm 9.2.4. Scoring and fixing is done in the same way as in the SFA heuristic: At first we determine for each lot-type and branch the best-fitting multiplicity. Best fitting in that sense means that there is no other multiplicity for which for the considered branch and lot-type a higher expected revenue can be achieved . Starting from this, for each branch we determine the three locally best fitting lot-types and add a score of 100 to the best fitting lot-type, a score of 10 to the second best fitting lot-type and a score of 1 to the third best fitting lot-types.

With this we have implicitly assigned a score to each lot-type $\ell \in L$, where most of the lot-types obtain the score zero. We can extend this scoring to the κ-subsets of L by summing up the individual scores, so that we implicitly get an order of the $\binom{|L|}{\kappa}$ many feasible lot-type combinations. With this we traverse the κ-subsets of L in descending order and break up if a predefined number of subsets is reached, where ties are broken arbitrarily. For this purpose we apply an approach which we adopted from the original SFA heuristic. It makes it possible to traverse the "most promising" lot-types without explicitly generating all such subsets beforehand. In the corresponding search tree a node at Depth i corresponds to the i-th lot-type in a subset. Each node/lot-type with ordering j according to the scoring has only childnodes with ordering $> j$, i.e. smaller or equal scores. Such we avoid to obtain permutations of the same subset of lot-types. At a leaf in depth κ we are given a complete κ-subset of lot-types. Bounding in the tree is possible. The maximum possible overall score $score^{\max}$ is given by the summed up single scores for the κ first lot-types in the ordering. The minimum possible overall score $score^{\min}$ by the summed up single scores for the κ last lot-types. We perform depth-first-searches in our search tree for the overall scores $score^{\text{cur}} = score^{\max}, score^{\max} - 1, \ldots, score^{\min}$ until a predefined number nr^κ of found κ-subsets of lot-types is reached. Because in a branch of the tree the lot-types are ordered decreasingly according to their scores we obtain an upper bound ubs for the overall score of all induced κ-subsets by adding κ minus depth times the score of the

current lot-type/node to the scores of the already fixed lot-types. Similarly we obtain a lower bound lbs: to the score of the fixed lot-types we add the κ minus depth smallest occurring scores. If $score^{\text{cur}} < lbs$ or $score^{\text{cur}} > ubs$ we can prune the current branch of the search tree.

In the fixing step we assume that the applicable lot-types are restricted to the current κ-subset of L. Now we differ from the original SFA heuristic as described in Chapter 2. In our implementation we solve the SLDP(W^E) for the fixed lot-types directly via an MIP-solver – with a predefined solving time limit tb – and have not to perform an adjust step.

Notation 8 (SLDP$\left(W^E\right)$ for a subset $\hat{L}$ of lot-types)**.** We consider a subset $\hat{L} \subseteq L$ of lot-types. We denote the SLDP$\left(W^E\right)$ as formulated in Problem 7 restricted on the set $\hat{L}$ of lot-types by SLDP$_{\hat{L}}\left(W^E\right)$.

Algorithm 16 ISPO-SF

Require: complete data of an instance of ISPO
lot-type revenues $\hat{a}^{e\to t}_{b,\ell,m}$ for all $b, \in B, \ell \in L$ and $m \in M$ and all maps $e \to t$ from the set W^E
lower bound $z_{\text{SLDP}(W^E)}$ for the objective value
Ensure: supply in terms of lots and corresponding expected revenue $z_{\text{SLDP}(W^E)}$
1: init $score(\ell) = 0\ \forall \ell \in L$
2: **for all** $b \in B$ **do**
3: **for all** $\ell \in L$ **do**
4: determine the revenue maximizing multiplicity $m(\ell, b) = \arg\max_{m\in M} \hat{a}^{e\to t}_{b,\ell,m}$
5: **end for**
6: determine the three best fitting lot-types ℓ_1, ℓ_2, ℓ_3
$\ell_1 = \arg\max_{\ell\in L} \hat{a}^{e\to t}_{b,\ell,m(\ell,b)}$, $\ell_2 = \arg\max_{\ell\in L,\ell_2\neq\ell_1} \hat{a}^{e\to t}_{b,\ell,m(\ell,b)}$,
$\ell_3 = \arg\max_{\ell\in L,\ell_3\neq\ell_1,\ell_3\neq\ell_2} \hat{a}^{e\to t}_{b,\ell,m(\ell,b)}$
7: update the scoring $score(\ell_1) = score(\ell_1) + 100$, $score(\ell_2) = score(\ell_2) + 10$, $score(\ell_3) = score(\ell_3) + 1$
8: **end for**
9: **while** a predefined number nr^κ of passed through κ-subsets of lot-types is not reached **do**
10: consider the not yet considered $\kappa-$subset $\hat{L}$ with the highest overall score $\sum_{\ell\in\hat{L}} score(\ell)$
11: solve SLDP$_{\hat{L}}(W^E)$ with time limit $tb \to$ objective function value $z_{\hat{L}}$SLDP $\left(W^E\right)$ and related supply $x_{\text{SLDP}_{\hat{L}}(W^E)}$
12: **if** $z_{\text{SLDP}_{\hat{L}}(W^E)} > z_{\text{SLDP}(W^E)}$ **then**
13: $z_{\text{SLDP}(W^E)} = z_{\text{SLDP}_{\hat{L}}(W^E)}$
14: $x_{\text{SLDP}(W^E)} = x_{\text{SLDP}_{\hat{L}}(W^E)}$
15: **end if**
16: **end while**
17: **return** $z_{\text{SLDP}(W^E)}$ and $x_{\text{SLDP}(W^E)}$

9.2.5 Solving the POP

Our approach for solving the price optimization stage is outlined in Algorithm 17, ISPO-PO. We use Algorithm 3 – POP-DYN – to solve the Price Optimization Problem for each particular scenario from the set E to optimality. From the the expected revenue over all these scenarios we have to subtract the corresponding lot-opening costs and the acquisition price for the supplied items. We take over these costs from the preceding solution of the size optimization stage.

9.2.6 Computational results

We apply ISPO-PingPong also to all instances of the test set $\mathcal{I}_6^{\text{test}}$. We set $nr^\kappa = 100$ and $tb = 60$ seconds. We perform four different ways of fixing the price trajectories to

Algorithm 17 ISPO-PO

Require: complete data of an instance for ISPO
supply x in terms of lots
Ensure: optimal price trajectory for each scenario according to x
1: compute overall acquisition, lot-opening and handling costs c^- for the given supply x.
2: set $z^* = c^-$
3: **for all** scenarios $e \in E$ **do**
4: perform price optimization, Algorithm 3, POP-DYN according to the supply $x \rightarrow$ revenue $a^*(e)$
5: $z^* = z^* + \text{Prob}(e) \cdot a^*(e)$
6: **end for**
7: **return** z^*

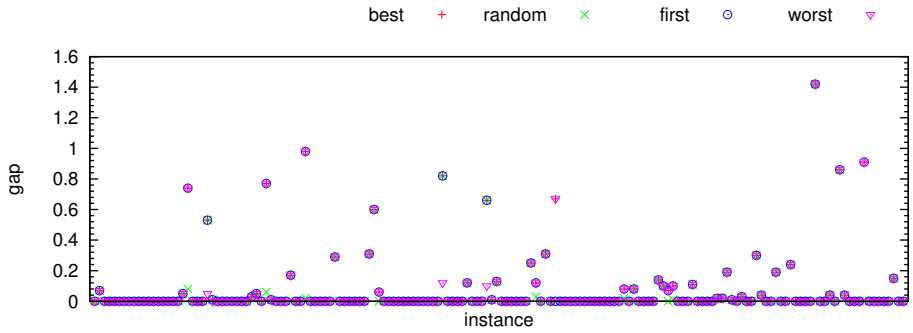

Figure 9.5: ISPO-PingPong – goodness of solutions

the particular scenarios at the beginning of the algorithm: For each scenario e we fix the price trajectory t

1. that yields the highest optimal objective function value of the related single-supply-relaxation SLDP-CB($\{e \rightarrow t\}$), "best" bound,

2. randomly,

3. without mark-downs during the real sales process (first in the depth-first generation of the price trajectories, see the enumeration tree in Section 4.2), "first" bound,

4. that yields the worst optimal objective function value of the SLDP-CB($\{e \rightarrow t\}$), "worst" bound.

We first take a look at the optimality gaps that are depicted in Figure 9.5.

Overall we get averagely an optimality gap of 0.078% for fixing the price trajectories with the best bounds, a gap of 0,055% for fixing the price trajectories randomly and a gap of 0,074% for choosing the first price trajectory. For fixing the worst price trajectory the resulting gap is 0.056%. That means in these cases we could achieve better results by starting with supposed "not promising" price trajectories. But actually all gaps are tiny enough to justify each of the outlined setting for fixing the price trajectories.

With the choice of supposed "better" trajectories we soon arrive at a suboptimal local maximum. This is approved by the results depicted in Figure 9.6. Already at first glance we can see that the strategy with the smallest gap – the randomly and worst choice of the price trajectories – needs the most iterations in all cases, while fixing the best trajectory yields a smaller number of iterations. We always counted a half iteration

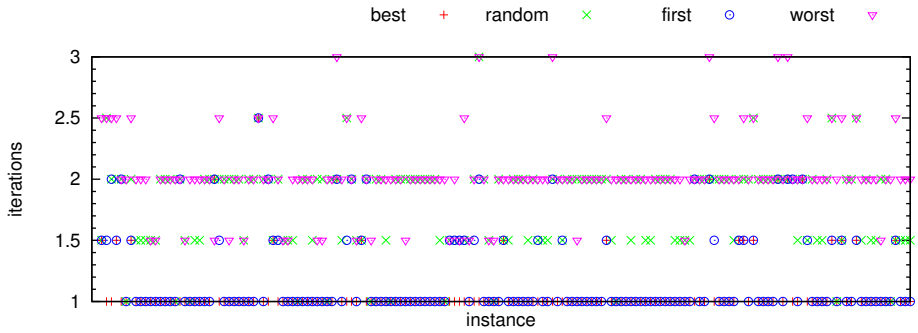

Figure 9.6: ISPO-PingPong – number of iterations

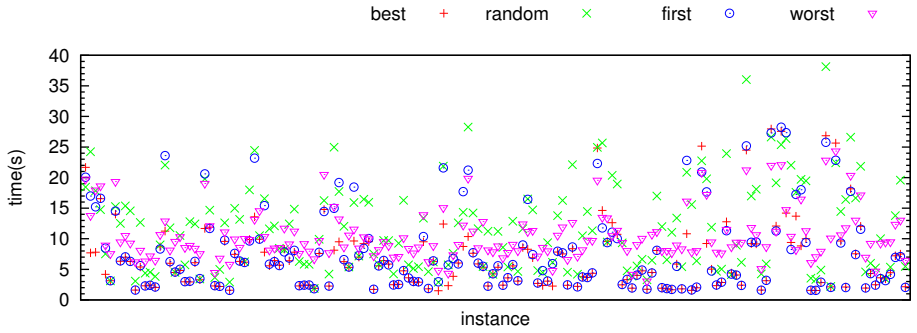

Figure 9.7: ISPO-PingPong – runtime

for performing size and price optimization. Averagely we get 1.59 iterations for fixing the best trajectories, 2.29 iterations for fixing the trajectories randomly or worst and 1.69 iterations for the price trajectory without a mark-down.

This behavior is also recognizable in the runtime, see Figure 9.3: averagely 6.85 seconds for setting 1, 11.86 seconds for setting 2, 7.84 seconds for setting 3 and 8.70 seconds for the last setting.

Results for other choices of nr^{κ} and tb are depicted in Appendix A.

9.2.7 Similarities to familiar approaches

At this point we want to remark that the main aspect of ISPO-PingPong – alternating size and price optimization until convergence – can also be connected to some general approaches in literature. The principle of our heuristic is similar to the principle of evolutionary algorithms, see for example [Mie99]. The idea of evolutionary algorithms is to assign a so-called *fitness*-function to the solutions of the problem and iteratively in a *selection*-step to combine the best-solutions to get solutions with higher *fitness*. This is done until convergence. In our case the *fitness* of the supply in terms of lot-types is given by the expected revenue the price optimization stage yields. By combining the local optimal supply with the local optimal mark-down strategy we possibly get a supply which results in higher revenue.

One could also connect the principle of ISPO-PingPong with the principle of bilevel programming. A bilevel program consists of an upper-level and a lower-level optimiza-

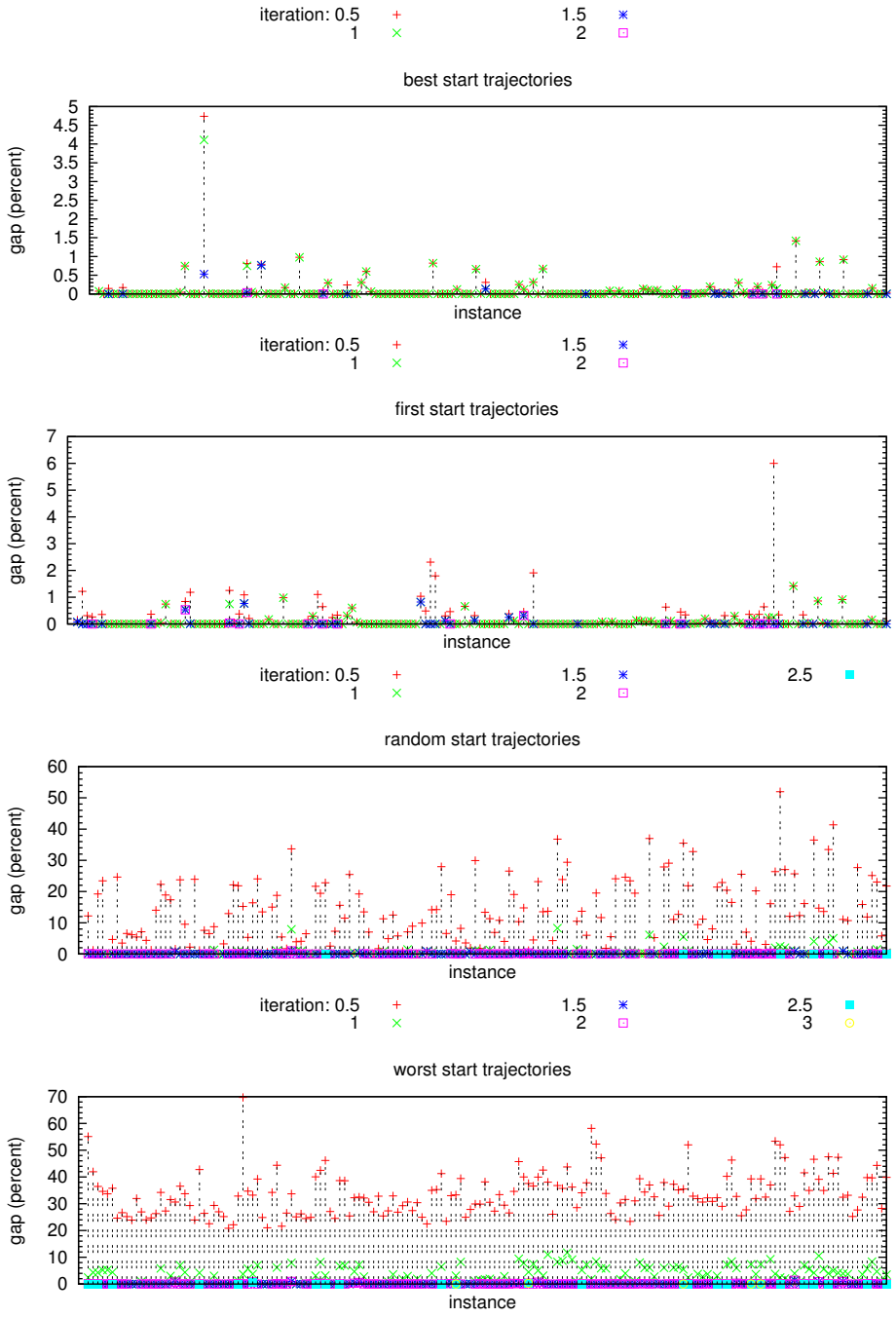

Figure 9.8: ISPO-PingPong – Progress of gaps

tion problem. The lower-level problem considers a variable x as a parameter to compute the optimal value of a variable y while the upper-level problem obtains the optimal value of x by using the value of y computed in the lower-level problem [CCMGB10]. In our case – by virtue of reversible recourse – we can see the size optimization stage and also the price optimization stage as both, as upper-level and as lower-level subproblems.

9.3 Computational results for real-world instances

We now compare ISPO-BAB with ISPO-PingPong for real instances from the set $\mathcal{I}$, Appendix E. We applied ISPO-BAB by using combinatorial bounds CB, extended combinatorial bounds ECB and extended LP bounds ELPB; LP bounds LPB are only applied in depth 1 of the Branch&Bound tree. Moreover, in a preprocessing step the state space of valid price trajectories is restricted by dominance DOM, see Algorithm 10. We performed ISPO-PingPong for two different settings:

1. $nr^{\kappa} = 1$, $tb = 20$ seconds,
2. $nr^{\kappa} = 100$, $tb = 60$ seconds.

In the second column in Table 9.3 "non.dom.(%)" we state the number(percentage) of remaining leaves in the enumeration tree after applying the dominance check for the price trajectories. By checking dominance most of the price trajectories can be excluded a priori , averagely more than 99.995 percent and so the number of leaves in the Branch&Bound tree reduces extremely.

In the third column we state the number(percentage) of the solved SLDP(W^E)s "#ILP(%)". The percentages in the fourth and the subsequent columns are related to the number of remaining leaves we state at column "non.dom". We see that in relation only a small number of all possible SLDP(W^E)s has to be solved. If we would relate the number of solved SLDP(W^E)s to the number of leaves of the hole enumeration tree without the restriction to non dominated price trajectories, the maximum number of 104 solved SLDP(W^E)s for Instance 2 would mean that just 0.0002156% of all possible SLDP(W^E)s had to be solved. In the next seven columns for all applied dual bounds overall solving time "t..." in seconds "(s)" or hours "(h)" and the number "#..." of the computed related bounds are stated. Because we compute combinatorial bounds for all scenarios and price trajectories a priori, i.e. 100% of these bounds, we did not state this specific number. Let us take look at the computation times for the different bounds. We therefore take the average computation times in the last line of the table and divide it by the related average number of solved bounds. The computation time for combinatorial and extended combinatorial bounds in relation to (extended) LP bounds is very small: For one ECB averagely it amounts to about 0.14 seconds while the average time needed for computing the LP bound amounts to nearly one hour and for the extended LP bound to about 9 minutes. But for LP bounds and extended LP bounds in relation to the solution of the binary program SLDP(W^E) the computational effort is yet small. The overall computation time minus the time for all bounds except CB yield the approximate average computation time for the SLDP(W^E)s. It is more than 110 hours. (The time for traversing the nodes in the tree is neglectable.)

Not every SLDP(W^E) is solved until the internal Branch&Bound process of the MIP solver starts. In Table 9.3 we did not distinguish between SLDP(W^E)s solved up to optimality and SLDP(W^E)s for which only the LP relaxation is solved. Because we

hand over the current best bound to the MIP solver it may be that the solving process breaks up earlier. Moreover, because the ELPB at depth $|E|$ equals the LP relaxation of the related SLDP(W^E) we do not compute the bound ELPB explicitly at depth $|E|$. So the columns "#ELP(%)" and "tELP(h)" in our case of three scenarios are only related to depth 2.

At the columns with label "ILP*(%)" and "t*" we state the number of SLDP(W^E)s that have to be solved at the leaves and the solving time until the optimal solution was found. We see that mostly the optimal solution is found very fast after averagely 5.9 solved SLDP(W^E)s. On the other hand altogether averagely 64.3 SLDP(W^E)s have to be considered. This speaks for our preprocessing step in which for the particular scenarios the price trajectories according to their single supply relaxations are ordered.

The results of ISPO-PingPong for the two settings are stated in the last six columns. We state the solving time in minutes "t/min", the number of iterations "#iter" and the optimality gap "gap(%)". Also with the first setting – i.e. at every iteration of the size optimization stage ISPO-SF only one SLDP($W^{E'}$) is solved – we mostly can get proper results. For the second setting where we traverse the 100 "best" κ-combinations we get no gap higher than 0.69%. The average solving time amounts to 12.77 minutes.

i	ISPO-BAB												ISPO-PingPong1			ISPO-PingPong2		
	t/h	non.dom.	#ILP(%)	tCB(s)	#ECB(%)	tECB(s)	#LP(%)	tLP(h)	#ELP(%)	tELP(h)	ILP*(%)	t*/h	t/min	#iter	gap(%)	t/min	#iter	gap(%)
1	47.75	1331(0.0057)	81(6.09)	56.69	101(6.96)	8.44	5(15.15)	2.68	21(17.36)	2.46	2(0.15)	9.60	1.05	2	0.30	22.41	2	0.08
2	88.64	1331(0.0057)	104(7.91)	56.58	123(8.47)	9.55	4(12.12)	3.94	20(16.53)	3.82	30(2.25)	45.80	1.01	1.5	0.26	12.61	2	0.01
3	130.93	1331(0.0057)	85(6.39)	89.36	113(7.78)	19.00	5(15.15)	5.68	29(23.97)	6.19	23(1.73)	46.21	1.05	1.5	0.65	20.84	1.5	0.09
4	214.72	1331(0.0057)	90(6.76)	87.76	110(7.58)	22.86	5(15.15)	5.66	21(17.36)	4.98	2(0.15)	12.61	1.06	2	0.31	24.67	1.5	0.06
5	32.58	1331(0.0057)	34(2.55)	57.32	48(3.31)	3.55	4(12.12)	2.21	15(12.40)	1.92	2(0.15)	29.54	0.94	1	0.69	6.50	1	0.69
6	1.29	1(0.0000)	1(100)	56.96	0(0.00)	0.00	0(0.00)	0.00	0(0.00)	0.00	1(100)	1.23	0.93	1	0.50	6.47	1	0.50
7	51.27	1(0.0000)	1(100)	69.34	0(0.00)	0.00	0(0.00)	0.00	0(0.00)	0.00	1(100)	51.25	1.08	1	0.90	8.55	1	0.18
8	4.97	1331(0.0057)	28(2.10)	50.65	44(3.03)	2.33	4(12.12)	2.11	17(14.05)	0.36	1(0.08)	0.19	0.88	2	1.32	29.56	1.5	0.20
9	3.91	1331(0.0057)	21(1.58)	50.26	35(2.41)	1.75	4(12.12)	1.98	15(12.40)	0.35	1(0.08)	0.18	0.86	1.5	1.68	11.59	1	0.29
11	12.53	1331(0.0057)	58(4.36)	52.58	83(5.72)	5.28	5(15.15)	1.34	26(21.49)	0.81	3(0.23)	0.94	0.86	1.5	1.78	29.73	1.5	0.23
12	10.84	1331(0.0057)	47(3.53)	52.38	68(4.68)	4.35	5(15.15)	1.54	22(18.18)	0.71	1(0.08)	0.65	0.95	1.5	1.69	12.11	1	0.20
13	10.26	1331(0.0057)	45(3.38)	52.06	64(4.41)	4.00	5(15.15)	1.43	20(16.53)	0.67	1(0.08)	0.63	0.94	1.5	1.83	8.74	1	0.21
44	112.69	1331(0.0057)	92(6.91)	56.92	112(7.71)	9.11	5(15.15)	2.56	21(17.36)	2.45	2(0.15)	9.51	1.02	1	1.89	12.95	1	0.23
45	123.71	1331(0.0057)	81(6.09)	58.07	113(7.78)	9.08	5(15.15)	2.62	33(27.27)	3.60	29(2.18)	62.55	1.11	2	0.26	12.43	2	0.00
46	197.57	1331(0.0057)	80(6.01)	89.76	112(7.71)	18.22	5(15.15)	5.78	33(27.27)	7.00	1(0.08)	6.23	0.99	1	0.26	6.66	1	0.00
48	67.77	1331(0.0057)	53(3.98)	58.03	71(4.89)	5.79	4(12.12)	2.34	19(15.70)	2.26	2(0.15)	7.21	1.09	1.5	0.52	12.37	1.5	0.52
49	4.58	1(0.0000)	1(100)	93.19	0(0.00)	0.00	0(0.00)	0.00	0(0.00)	0.00	1(100)	4.56	0.99	1	0.16	7.71	1	0.16
50	67.59	1331(0.0057)	45(3.38)	88.87	65(4.48)	9.98	4(12.12)	5.39	21(17.36)	4.98	1(0.08)	4.55	1.00	1	4.12	8.05	1	0.22
52	4.58	1(0.0000)	1(100)	90.09	0(0.00)	0.00	0(0.00)	0.00	0(0.00)	0.00	1(100)	4.55	0.99	1	4.07	7.93	1	0.21
53	340.49	1331(0.0057)	129(9.69)	90.78	159(10.95)	25.22	5(15.15)	6.84	31(25.62)	6.96	43(3.23)	125.05	1.09	1.5	0.11	14.16	1.5	0.00
61	237.59	1331(0.0057)	90(6.76)	92.71	119(8.20)	20.24	6(18.18)	13.57	30(24.79)	7.20	1(0.08)	4.72	1.12	1.5	0.36	11.99	1.5	0.05
62	291.86	1331(0.0057)	102(7.66)	91.61	129(8.88)	23.25	5(15.15)	6.21	28(23.14)	6.33	1(0.08)	4.61	1.01	1	0.33	6.80	1	0.14
63	296.24	1331(0.0057)	116(8.72)	91.70	141(9.71)	23.81	5(15.15)	4.59	26(21.49)	4.96	2(0.15)	10.74	1.11	2	0.19	13.77	1.5	0.12
64	328.68	1331(0.0057)	118(8.87)	91.19	143(9.85)	26.45	5(15.15)	5.80	26(21.49)	5.80	1(0.08)	4.73	1.11	2	0.20	7.60	1	0.12
65	1.17	1(0.0000)	1(100)	91.29	0(0.00)	0.00	0(0.00)	0.00	0(0.00)	0.00	1(100)	1.17	0.99	1	0.25	7.65	1	0.18
66	154.40	1331(0.0057)	106(7.96)	91.51	129(8.88)	23.54	5(15.15)	5.83	24(19.83)	5.45	2(0.15)	9.32	1.12	2	0.25	14.15	2	0.17
67	323.95	1331(0.0057)	125(9.39)	90.88	156(10.74)	29.11	6(18.18)	9.86	32(26.45)	5.97	2(0.15)	9.37	1.0	1	0.33	6.77	1	0.15
∅	117.13	1085(0.0046)	64.3(23.34)	74.02	82.90(5.71)	11.29	4.00(11.90)	3.70	20.00(16.22)	3.16	5.9(18.95)	17.32	1.01	1.43	0.93	12.77	1.30	0.19

Table 9.1: ISPO-BAB and ISPO-PingPong applied on real instances

9.4 General goodness of ISPO-PingPong

In the previous section we showed that the optimality gap for our heuristic for real-world instances averagely amounts to 0.19% and in the worst case to 0.69% depending on the setting. Now we apply the heuristic on a small example to show that we cannot guarantee such small gaps in general. As in our accompanying example we consider only two branches $B = \{1, 2\}$ and two sizes $S = \{\mathrm{S}, \mathrm{L}\}$. Only one lot-type is allowed for supplying the two branches, i.e. $\kappa = 1$. We consider all lot-types with at minimum one item per size, i.e. $v^{\min} = 1$ and at maximum two items per size, $v^{\max} = 2$. We only allow lot-types with cardinality 3. This yields the lot-types $(1, 2)$ and $(2, 1)$. The set of multiplicities is given by $M = \{1, 2, 3\}$. We set $\underline{I} = 0$ and $\overline{I} = 20$. In this example we assume that pick-cost and lot-opening costs take value zero. We consider four sales periods including the sellout period $k_{\max} = 3$. The discounting factor is also set to zero, $\rho = 0$. Moreover we set the fixed and variable mark-down costs to zero, i.e. $\mu_v = \mu_f = 0$. We are given four prices including the salvage value. It is $P = \{0, 1, 2, 3\}$ with $\pi_0 = 10, \pi_1 = 9, \pi_2 = 8$ and $\pi_3 = 4$. We number the price trajectories t according to the following table:

index	t
1	$(0, 0, 0, 3)$
2	$(0, 0, 1, 3)$
3	$(0, 0, 2, 3)$

For simplicity's sake we consider only one scenario with probability one. The demands per period k and price index p are given as stated in the following tables.

$(1, \mathrm{S})$

k/p	0	1	2
0	1.0	-	-
1	0.0	-	-
2	0.0	3.0	3.1

$(1, \mathrm{L})$

k/p	0	1	2
0	1.0	-	-
1	1.0	-	-
2	0.0	1.1	3.0

$(2, \mathrm{S})$

k/p	0	1	2
0	1.0	-	-
1	1.0	-	-
2	0.0	1.1	1.1

$(2, \mathrm{L})$

k/p	0	1	2
0	1.0	-	-
1	0.0	-	-
2	0.0	0.0	0.0

We state the single supply revenues computed by Algorithm 6 in the table found below. In the last column we are given the additional revenue for the demand-exceeding numbers of items according to Observation 2. Because we are given no mark-down costs, the additional revenue always amounts to $\pi_{k_{\max}} - \mathrm{ap} = 5 - 4 = -1$.

(branch,size)	t/n	1	2	3	4	5	6	6+
(1,S)	1	5.0	4.0	3.0	2.0	1.0	0.0	−1.0
	2	5.0	9.0	13.0	17.0	16.0	15.0	−1.0
	3	5.0	8.0	11.0	14.0	13.4	12.4	−1.0
(1,L)	1	5.0	10.0	9.0	8.0	7.0	6.0	−1.0
	2	5.0	10.0	14.0	13.5	12.5	11.5	−1.0
	3	5.0	10.0	13.0	16.0	19.0	18.0	−1.0
(2,S)	1	5.0	10.0	9.0	8.0	7.0	6.0	−1.0
	2	5.0	10.0	14.0	13.5	12.5	11.5	−1.0
	3	5.0	10.0	13.0	12.4	11.4	10.4	−1.0
(2,S)	1	5.0	4.0	3.0	2.0	1.0	0.0	−1.0
	2	5.0	4.0	3.0	2.0	1.0	0.0	−1.0
	3	5.0	4.0	3.0	2.0	1.0	0.0	−1.0

At the beginning of ISPO-PingPong a price trajectory for each scenario is fixed. In this example at first we choose the "first" price trajectory 1. For this trajectory the best supply in terms of lot-types is computed. Because $\kappa = 1$ we can only choose one lot-type, either $(1, 2)$ or $(2, 1)$, for supply.

According to the single supply revenues an optimal supply for Price trajectory 1 is given by choosing Lot-type $(1, 2)$ in Multiplicity 1 for Branch 1 and in Multiplicity 2 for Branch 2. This yields a revenue of 27.

The next step is to reject the fixed price trajectory and if possible to choose one which is optimal for the given supply. Otherwise the heuristic converges.

Adding up the corresponding single supply revenues yields that no other price trajectory is better for the given supply. For the price trajectories 2 and 3 the revenues for the given supply also amount to 27. The approach converges with objective function value 27.

An alternative would be to start with the "best" price trajectory – that means the price trajectory t which yields for our scenario e the highest objective function value of the single supply relaxation SLDP-CB($\{e \to t\}$). This is Price trajectory 3. The best revenue in terms of single supplies for this trajectory is 51 which is given by supply $(4, 5)$ for Branch 1 and $(3, 1)$ for Branch 2. Price trajectory 1 yields just an revenue of 30 with supply $(1, 2)$ for Branch 1 and $(2, 1)$ for Branch 2. The revenue for Trajectory 2 amounts to 50 and results from a delivery of $(4, 3)$ for Branch 1 and $(3, 1)$ for Branch 2.

That means we start with Price trajectory 3 and compute the best supply in terms of $\kappa = 1$ chosen lot-type. The optimal supply for this trajectory is given by supplying 3 times Lot-type $(1, 2)$ to both branches. The resulting revenue amounts to 42.

Now we have to reject the current price trajectory again and to compute the best trajectory in terms of the current supply. According to the single supply revenues this is also Price trajectory 3. For Trajectory 1 the revenue amounts to 18 and for Trajectory 2 it amounts to 38.5. That means the approach converges with a revenue of 42.

Solving ISPO for this example to optimality yields an optimal solution value of 46.50 which results from supplying Lot-type $(2, 1)$ three times to Branch 1 and two times to Branch 2. Price trajectory 1 is optimal.

For this example neither using the "best" trajectory nor the "first" trajectory – which is also the "worst" – as start trajectories lead to a solution with small gaps. The gaps amount 41.94% for the "first" and 9.68% for the "best" trajectory.

Thus, in general we can not assume such a good performance as on our instances.

We want to state some specialties of our real-world instances: For our instances the range between the lower and upper bound as a rule amounts to about maximal 10% of $\underline{I} + \frac{\overline{I}-\underline{I}}{2}$, see Remark 1. In this example it is much higher, namely 200%. Moreover in the real-world case the deviation among the demands is with mostly zero or one sold items less.

Although at this point we can not evidence a general warranty of goodness for ISPO-PingPong, the bounds in terms of our real-world instances are small enough to justify practical use.

9.5 Conclusion of the chapter

We presented the Branch&Bound solver ISPO-BAB which solves the Integrated Size and Price Optimization Problem for all tested instances to optimality. We branch on maps "scenario to price trajectory". Dual bounds are obtained by extensions of the wait-and-see solution from stochastic programming.

For practical use at our industrial partner we propose the heuristic ISPO-PingPong. The principle is to alternate size and price optimization until convergence. This is possible because of the special structure of ISPO – the reversible recourse. We show that ISPO-PingPong for the tested real-world instances yields solutions with an average optimality gap of 0.19% – the maximum gap amounts 0.69% – in averagely 12.77 minutes.

Chapter 10

DISPO in practical application – real-world experiments

The collaboration with our industrial partner gave us the opportunity to test the practical relevance of DISPO in a real-world field study.

With real-world experiments we want to verify that DISPO performs better than the method currently in use at the partner, i.e. the LDP together with a manual determination of mark-downs. For that reason the DISPO-team performed a so-called *single-blind experiment* where test and control branches compete against each other. We give some basics about statistical experiments in Section 10.1 before we apply them in Section 10.2 to our field studies.

During the cooperation also field studies only in terms of price optimization were performed. In Section 10.3 we will outline the main results. We show how heavily mark-downs can directly affect the number of sales and that price optimization can also increase the realized revenue.

Because performing a field study is expensive in terms of work and money we estimated the potential of improvement a change from LDP to ISPO-based supply would bring along. The results – which we outline in Section 10.4 – convinced our partner and the DISPO-team to perform a five-month field study.

With DISPO we could finally increase the realized revenue about more than 1.5 percentage points. Moreover by regarding the field study as a statistical experiment we can give a statement about the significance of the result. We present the details in Section 10.5.

10.1 Performing statistical experiments

With real-world experiments we want to figure out if DISPO – or also price optimization as a part of DISPO – performs better than the methods currently implemented at our partner. Moreover we want the results to be statistically significant.

In order to apply statistical methods later on we want to introduce some statistical basics in this section. We start with a classification of blind experiments in Subsection 10.1.1. We outline the term statistical significance in Subsection 10.1.2. To make a point about statistical significance we have to perform a test of significance. We mention the most common tests in Subsection 10.1.3 before we focus on the *Wilcoxon signed-rank test* in Subsection 10.1.4.

In this section we are mainly guided by [FPP07], [Raj06] and [Kan06].

10.1.1 Blind experiments

A blind experiment is a statistical experiment where not all people involved are informed about certain aspects to avoid bias.

Blind experiments are typically applied in medical tests. The group of probands is divided into a test and a control group where the test group gets the medicament and the control group just a placebo. If one wants to examine the effect of a medicament it is usual that the experimentees are not informed in which group they are. Otherwise the experimentees might be affected by this information. If all other involved persons – except the experimentees – have full information about the categorization we talk about a *single-blind experiment*.

In some cases it is useful that also the researchers do not know about the category of the tested persons. They might treat the probands accordingly. In this case we would talk about a *double-blind experiment*.

10.1.2 Statistical significance

If our new method performs worse or better than the method currently in use we want to state how big the role of chance for this result was. If the probability that the result could be caused by pure chance is not small enough we would not give general statements about a better or worse performance. The so-called *null hypothesis* says that the method leads to no differences (or also to no better/worse performance). An *alternative hypothesis* that it does. A test is *statistically significant* if the probability that its outcome is the result of chance is smaller than a predefined *significance level*. A common choice for the significance level is 5%. To make a point about statistical significance a so-called *test staticstic* is computed. A test statistic is defined as a measurement of the difference between the data and the statement of the null hypothesis, [FPP07]. The test statistic follows a *test distribution*. If the probability to obtain the test statistic under the test distribution – the so-called *p-value* – is smaller than the significance level – then we call the result *statistically significant* – we reject the null hypothesis and rely on the alternative hypothesis. Rejecting the null hypothesis does not mean that the alternative hypothesis is true. It is only an evidence that the result is not caused by random.

10.1.3 Statistical tests in general

To evidence statistical significance there are several statistical tests. There are tests for *related* samples and tests for *unrelated* samples. A sample is called unrelated if groups of different individuals are compared. For related samples we compare groups with individuals related pairwise to an individual from the other group. Which test can be applied also depends on the kind of the data: Are the observations nominal, ordinal or given by a distribution? *Parametric* tests assume a specific distribution while *non-parametric* tests do not.

We distinguish between *two-sided* and *one-sided* tests. A two-sided test considers both, a better and a worse performance of the test sample simultaneously. The null-hypothesis says that both methods perform the same way, the alternative hypothesis that they do not. With a one-sided test we are only interested in a better/worse performance of the “new method”. The null-hypothesis says that it performs not better/not worse,

the alternative that it does. Because in our case we are interested in a worse or better performance we focus on one-sided tests in the following.

A common approach for two unrelated normal distributed samples is Student's *t-test*. The t-test compares differences between the means of the particular samples and compares them with the corresponding standard error to determine if the two samples arise from the same distribution. For related normal distributed samples the t-test can also be applied in a similar way. For further information see for example [FPP07].

In the case that no distribution for the observations can be assumed (but also for observations from a specific distribution) non-parametric tests can be applied. The tests for non-parametric ordinal data assign ranks to the observations. As test statistic *rank-sums* are computed. The test distribution is the distribution of the rank-sums.

For unrelated samples the Mann-Whitney test is commonly used. At first all observations are ordered increasingly and ranks are assigned in terms of the ordering. Then, by summing up all ranks of one sample the rank sum – the test statistic – is computed. To determine the role of random one computes the p-value as the probability to get the observed rank-sum (or for one-sided tests the observed rank-sum or a higher/lower one) among all other possible rank-sums.

For related samples there is a similar approach named Wilcoxon signed-rank test.

10.1.4 Wilcoxon signed-rank test

In order to certify statistical significance for two related ordinal samples Wilcoxon signed-rank test is very common. The test is named after Frank Wilcoxon who presented it together with the rank sum test for non-paired observations also called Mann-Whitney test in [Wil45]. Wilcoxon signed-rank test is an alternative to the Student's t-test if no normal distribution can be assumed. It yields a statement about the symmetric distribution of the pair differences around the median.

The test can be performed as one-sided or two-sided test. For our purposes only the one-sided test is relevant. Therefore we formulate Wilcoxon signed-rank test as one-sided test.

It is checked if the differences of the ordered paired observations (test − control) are distributed symmetrically around or right of the median $\tilde{x}$ or symmetrically around or left of the median $\tilde{x}$. Thus, the null hypothesis in the first case is

$$H_0 : \tilde{x} \leq 0, \tag{10.1}$$

and the alternative hypothesis

$$H_1 : \tilde{x} > 0. \tag{10.2}$$

In the second case, the null-hypothesis is

$$H_0 : \tilde{x} \geq 0, \tag{10.3}$$

and the alternative hypothesis

$$H_1 : \tilde{x} < 0. \tag{10.4}$$

In the first case the null hypothesis is equivalent to the statement that the distributions of the paired observations for the two samples are identical or that the distribution of the test sample is shift to the left. In the second case that they are identical or that the distribution of the test sample is shift to the right.

We now describe the different steps for performing the Wilcoxon signed-rank test. They are illustrated on the following small example.

We assume our observations for the test and the control sample are given as stated in the subsequent table.

observation	1	2	3
test	0.45	0.58	0.63
control	0.52	0.55	0.46

1. Computing the differences of the paired observations

We compute the differences of the observations for each test-control pair. This yields:

observation	1	2	3
difference	-0.07	0.03	0.17

2. Ordering the absolute values increasingly and assigning ranks

The next step is to order the absolute values of the differences from above increasingly. Additionally we store the sign of each difference: It says if the value for the observation from the test sample was higher or lower than the related observation from the control sample. The observations get ranks according to the ordering.

observation	2	1	3
abs. diff.	0.03	0.07	0.17
sign	$+$	$-$	$+$
rank	1	2	3

If absolute differences for test-control pairs are equal we talk about *ties*. If for example the three signed differences -0.03, 0.03 and 0.07 were observed there would be no obvious ranking. Rank 1 or rank 2 could be assigned to both of the two first differences. In this case it is common to assign the mean of the ranks the observations would occupy. In our case the first and the second observation get rank $\frac{2+1}{2} = 1.5$ while the third observation gets rank 3.

3. Computing the test-statistic – the rank-sum

Now the rank sum as test-statistic is computed by adding up all ranks with an associated difference with positive sign. If we want to check if the test distribution against the control distribution is shift to the right (as alternative hypothesis), with n test-control pairs the null-hypothesis is equivalent to that case that the rank sum is $\frac{n(n+1)}{4}$ or lower. If we are interested in a left shift of the test distribution the null-hypothesis is equivalent to that case that the rank sum is near $\frac{n(n+1)}{4}$ or higher. (It is $\frac{n(n+1)}{2} = \sum_{i=1}^{n} i$ the sum of all ranks according to the Gaussian sum.)

In our example the rank sum is $1 + 3 = 4$.

4. p-value

With the rank-sum as test statistic and the distribution of the rank sums as test distribution we are now able to compute the p-values for both one-sided tests.

In the first case – the test of a shift to the right for the test distribution – this is the probability of getting the observed rank-sum or a higher one by chance. In the case of checking a shift to the left we compute the probability of getting the observed rank-sum or a smaller one by chance.

For our small example this is easily done. We just have to compute the relative frequency of the observed rank sum and higher/smaller rank-sums among all possible rank-sums.

We denote the probability for getting a rank-sum of k or higher for a sample of test-control pairs with size n in the following by $P_n(X \geq k)$; the probability for getting a rank-sum of k or smaller by $P_n(X \leq k)$.

We consider all possible assignments of ranks and signs to our observations:

1	2	3	rank-sum
+	+	+	6
+	+	−	3
+	−	+	4
−	+	+	5
+	−	−	1
−	+	−	2
−	−	+	3
−	−	−	0

Our observed rank-sum for the example amounts to 4. The probability to get a rank-sum of 4 or higher than 4 by chance is $P_3(X \geq 4) = \frac{3}{8} = 37.50\%$. The probability for obtaining a rank-sum of 4 or lower by chance is $P_3(X \leq 4) = \frac{6}{8} = 75\%$.

In this example we cannot deny both null-hypothesises because both probabilities are greater than our significance level of 5%. Indeed the probability for randomly observing higher values for the test sample is smaller than the probability for randomly observing lower values. But both results are not significance. The probabilities that the results are caused by chance are too high.

For higher numbers of observations we can exploit the fact that the test statistic of the Wilcoxon rank-sum test can be approximated by the normal distribution, see for example [Mon10].

10.2 Performing our field-studies as statistical experiments

Now we want to apply the approaches outlined in the last section to real-world field-studies in terms of DISPO. In the first preliminary studies the DISPO-team performed, the industrial partner provided pairs of articles. These were similar products sometimes differing in color only. Randomly the DISPO-team categorized the set of articles in test and control articles. The control articles were in the hand of the industrial partner while the new methods were applied on the test articles: The DISPO team performed a test for related samples where the samples are the test and the control articles. Related are the pairs of articles provided by the partner.

But the differences between the popularity of the articles was so enormous that a reliable statement about a better or worse performance of the new method in terms of realized revenue was not possible. Comparability of the articles was not given.

To evaluate performance of different supply policies anyhow we developed methods to at least make a statement about performance in terms of the size compliance. This means that we measure how well the supply per branch and size meets the related demand. This is – similar to the empirical estimation, see Chapter 3, Section 3.2.1 – done by observing the sales for an article to the day until 50% of the overall supply over all branches and sizes are sold. Because nearly all items are sold at the end of the selling time, we only observe the products within this "50%-time frame" to be able to

observe differences for the particular branches and sizes. If then we detect that a size is sold comparatively fast then we can assume an undersupply of this size. It speaks for an oversupply if the items of the considered size stay comparatively long in a branch. To indicate such behaviors we developed measurements to compare the selling speed of the different sizes. One of them is the *normalized sales rate deviation*. It is based on the consideration that a size-compliant supply would yield small differences among the sales for the different sizes for the observed time frame. This difference is measured by the standard deviation. Because the exact day when 50% of the articles are sold out can not always be measured we scale the sales rates for the different sizes to a mean of 50%. Another measurement is the so-called *top-dog deviation*. The top-dop deviation measures the difference between the probability that a size s is sold out first and the probability that s is sold out last. If the supply was size-compliant both probabilities should be nearly equal. Significance is measured by Wilcoxon's rank-sum test. For further reading see [KKR12].

Because the important performance metric, the realized revenue – its maximization is the aim of our industrial partner – is not measurable by observing articles, the DISPO-team chose a different test set-up. Instead of categorizing the articles in test and control articles pairs of similar branches were provided by our industrial partner. In order to obtain statistically assessable results, our industrial partner grouped branches into pairs according to economic key figures, like the size of the stores and revenue. Whether a branch was assigned to be a test or a control branch in such a pair was decided randomly. We will benefit from this controlled test set-up because we can apply the Wilcoxon signed-rank test to check significance of the results. For our industrial partner this test set-up brings about a higher effort and additional costs. This is also the reason why the DISPO-team did not perform this kind of test set-up earlier. Because usually all branches were informed the same way about mark-downs etc. our partner now had to divide the information flow. Another aspect is that for example different prices for an article in neighbouring branches lead to confusion and maybe to loss of image with customers. Especially in terms of the supply there were some other features about the performance of the field study we will mention in Section 10.5. But overall from our point of view a classification in test and control branches was the best way to obtain convincing results: While the popularity of non-replenished fashion articles is a priori not predictable, the assignment test-control-pair is based on exact comparisons of economic key figures.

The field-studies are designed similar to a single-blind experiment. Neither the employees or customers of the test branches nor the ones of the control branches knew about their participation in a field study.

We test statistical significance by applying the Wilcoxon signed-rank test since there is no indication that the observations indeed follow a normal distribution (or any other distribution). In the following we compare relative sales or relative revenues. Relative sales mean that we divide the number of sales in a branch by the related supply. Relative revenue means dividing the realized revenue by the maximum possible revenue. Both concepts are described in detail later on. We order our test-control pairs according to the absolute differences in terms of the observed quantity. The ranks are assigned accordingly with a positive sign if the test branch wins – if the revenues or the sales are higher than in the control branch – and otherwise a negative sign. With n test-control pairs and a observed rank-sum of k the p-value for a better performance of the test branches is given by $P_n(X \geq k)$, for a worse performance by $P_n(X \leq k)$.

10.3 POP-RH in real-world studies

At this point we want to share the main results of two field-studies we performed to obtain informations about the performance of the price optimization with receding horizon POP-RH, see 2.5.4. Price optimization was performed every sales week again exploiting latest sales figures, i.e. the knowledge about the scenario in effect. The DISPO-team performed different experiments from which we want to sketch out the results of the two last ones where the test set-up described above was applied. While for the first outlined experiment we were mainly concerned with the evaluation we performed and evaluated the second experiment completely. At first we will show that mark-downs can have a significant influence on the sales. Then our focus lies on the realized revenue.

10.3.1 Performing price optimization with receding horizon – POP-RH

In the field studies in terms of price optimization both, the test branches and the control branches, were supplied by our industrial partner. The control branches were completely managed by our partner who performs the mark-downs there. Two weeks after sales start the DISPO-team/we began to perform the price optimization process in the test branches. This time is needed to assign the scenario in effect according to the observed sales figures.

To exploit always the latest sales figures, at the end of each week. the demand estimation according to the observed sales was updated, see Chapter 3, 3.2.6. Then, price optimization was applied to compute an updated price policy. If the optimal price trajectory suggested a mark-down in the following two time periods the industrial partner was advised to implement exactly this mark-down.

We want to remark at this point, that because of the high amount of data our project partner had to transfer to the DISPO team, the proposed mark-downs could not exactly be applied the week after the evaluation of the latest sales figures. The mark-downs were applied one week later. In detail the sales data was available for the DISPO-team between Monday morning and Tuesday evening. Then the in terms of latest sales figures optimal price trajectory was computed. The partner obtained the by the POP-RH proposed mark-downs at the latest on Tuesday evening. Depending on the sales start of the article the mark-downs were performed the week after on Tuesday or Friday.

The partner provided all transactions and additional data on a server. Additionally to the about 8 GB of data that was needed for empirical demand estimation at the beginning of the field studies every week 4 GB of new data which included latest transaction data were transferred.

10.3.2 Sales increase by mark-downs

From the beginning of November 2009 until the end of March 2010 the DISPO-team performed a field study in terms of price optimization at the industrial partner. Compared were 26 test branches with 26 control branches.

There was one specialty. Because the partner had to get rid of old articles – old means winter products – to create space for the spring articles the DISPO-team decided to include penalty costs in the objective of the POP (without mark-down costs).

For Period k the penalty costs per remaining item are given by

$$pen(k) := \begin{cases} 0.1\text{ap} \cdot 2^{\frac{1}{2}(k-\omega+6)} & k \in K : k \geq \omega - 6, \\ 0 & \text{otherwise.} \end{cases} \tag{10.5}$$

The penalty costs depend on the current period, the current stock and the acquisition price. It is ω the number of weeks for the product from the sales start until the end of the field study. Penalty costs only arise in the last 6 weeks of the experiment. Because of the factor $\frac{1}{2}$ in the exponent, the penalty costs which depend from 10% of the acquisition price double all two weeks. For each non-sold item at Period k these costs were added in the objective of the POP. The idea is that each remaining item uses space that is needed for new products. Because at that time the DISPO-team had no estimation of the cost that remaining items would cause this artificial penalization was used.

The remaining parameter setting is given as stated in Table 10.3 (from the line four)[1]. POP-RH is performed as described in 10.3.1.

For the reason to get unbiased results the DISPO-team restricted the complete test set of 3050 articles to 1037 articles which were supplied to all test and control branches (we also performed an evaluation for all articles, the main result does not distinguish).

While in the control branches for the 1037 articles only 980 mark-downs were performed including the penalty costs led to 1928 mark-downs in the test branches.

In Figure 10.1 the effect of mark-downs for a sample of 66 of the 1037 articles is recognizable. We see that 6 weeks before the field study ends the "bad sellers" start to boom. If we look for example at the article which is depicted by the green rhombus, we see that this product is mark-downed at sales week 12 to about 0.25 of the starting price. We see that the sales increase rapidly in the next two sales weeks. We made this observation also for other articles and different sales weeks, see Appendix B.

We observed an average sales rate of 84.6 for the control branches and 90.4 for the test branches, respectively. This result is highly significant (in terms of Wilcoxon signed-rank test) with a p-value of nearly 0.00%.

The observed increase of sales by applying POP-RH (with penalty costs for non-sold items) in contrast to manual decisions on mark-downs is significant.

Because of the artificial penalty costs in the objective at this point we are not able to make a statement about the influence on the realized revenue. For this purpose we performed another field study.

10.3.3 Earnings increase by mark-downs

To compare POP-RH with manual decision making about mark-downs also in terms of money we performed another field study. Analogously to the previously described experiment all articles were supplied by our industrial partner. From 30 pairs of comparable branches the categorization in test and control branch happened randomly.

For this experiment we tried to model the reality as exactly as possible: POP-RH based on realistic estimations on all occurring costs. Our partner estimated fixed and variable mark-down costs. The fixed mark-down cost amount to $\mu_f = 7.0$ while the variable mark-down cost μ_v per item lie between 0.12 and 0.21 depending on the commodity group. It is $q_{k_{\max}} = 2$, that means after the last real sales periods two additional mark-downs are assumed. The remaining parameters are as stated in Table 10.3.

[1] In order not to reveal company internals, we state all values/costs with respect to artificial but consistent monetary units.

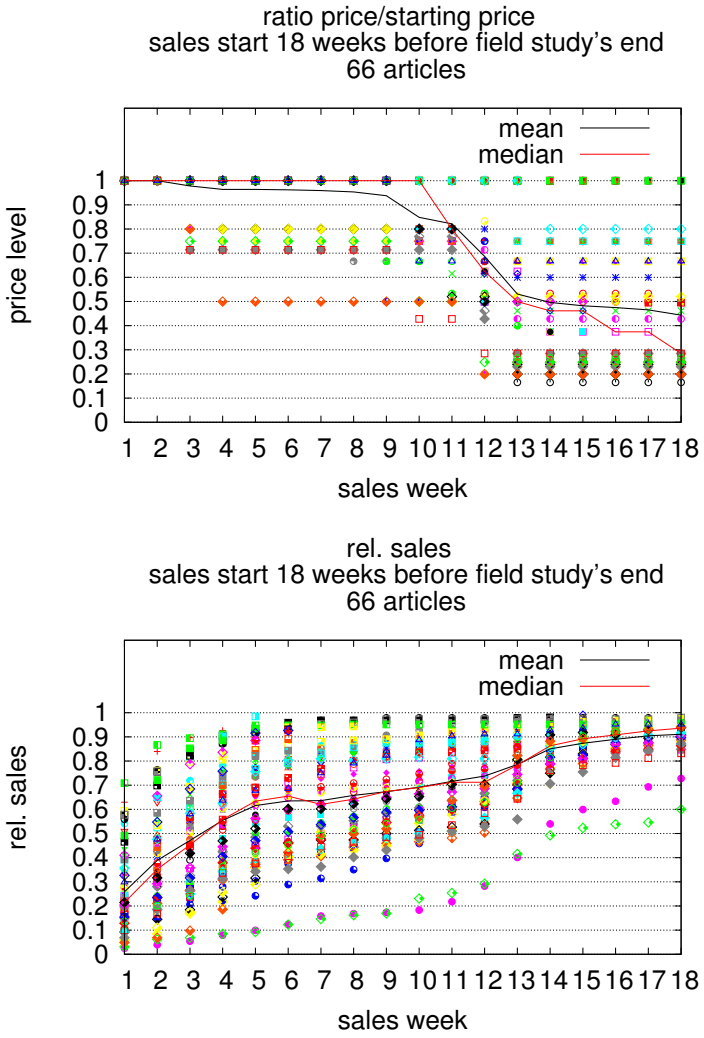

Figure 10.1: Effect of mark-downs

sample	relative realized objective	gross yield	sales
test	0.4774	0.6524	0.7891
control	0.4698	0.6494	0.7945

Table 10.1: Performance metrics – 2nd field study POP-RH

Originally there were 4668 products included in the test. To get possibly non-biased result our evaluation uses only 2298 from them which were supplied to all test and control branches.

We state some performance metrics for the test and control branches in Table 10.1.

We will focus on the *relative realized objective* which we will define in the following.

For each test-control pair of branches, the *realized revenues* over all articles in A were compared. That means, in particular, that expensive articles have a larger influence on the result than cheap articles. This point of view is in line with our partner's point of view.

For reasons of comparability we divide the realized revenue by maximum possible revenue in terms of the objective of the POP$^{\hat{e}}$.

This means for an initial stock $I^a_{b,s}$ for the considered branch b, size s and article a and a starting price π^a_0 we compute the *relative realized objective* of the mark-down decision $t^a = (t^a_0, t^a_1, \ldots, t^a_{k_{\max}})$ with $t = \left(t^a\right)_{a \in A}$ for Branch b as

$$RRO_{\mathrm{POP}}(b) = \frac{\text{objective of POP}^{\hat{e}} \text{ for } b \text{ achieved by } t}{\text{maximal possible objective}} = \frac{-\sum_{a\in\mathcal{A}}\sum_{s\in S} I^a_{b,s}\cdot \mathrm{ap}^a + \sum_{k\in K}\exp(-\rho k)\Big(\sum_{a\in\mathcal{A}}\sum_{s\in S}\hat{r}^a_{k,b,s} - \tilde{\mu}^a_k \hat{n}^a_k\Big)}{\sum_{a\in\mathcal{A}}\sum_{s\in S} I^a_{b,s}\cdot(\pi^a_0 - \mathrm{ap}^a)}. \quad (10.6)$$

Depending on Article a, ap^a denotes the acquisition price, π^a_0 the starting price and $I^a_{b,s}$ the initial stock per branch and size. During the sales process, we observed $\hat{r}^a_{k,b,s}$ (the realized yield for Branch b and Size s in Period k) for Article a and $\hat{n}^a_k$ (mark-down in Period k – yes or no).

Since we only consider a subset of branches we have to take into account that fixed mark-down costs must be scaled with respect to the number of considered branches. This way, we get mark-down costs $\tilde{\mu}^a_k$ for period k.

Because all test and control branches were supplied by our partner and now our focus lies on the price optimization stage we do not regard costs for supply in terms of lots as lot-opening costs and pick costs as they appear in ISPO.

We see that the mean relative realized objective in the test branches is about 0.76 percentage points higher than in the control branches. Also in terms of the other performance metrics POP-RH beats the manual price optimization. Yet, with a rank-sum of 285 and $P_{30}(X \geq 285) = 14.47\%$ and $P_{30}(X \leq 285) = 86\%$ Wilcoxon signed-rank test yields no significance. Still, the p-value for a better performance caused by pure chance is with 14.47% comparatively small.

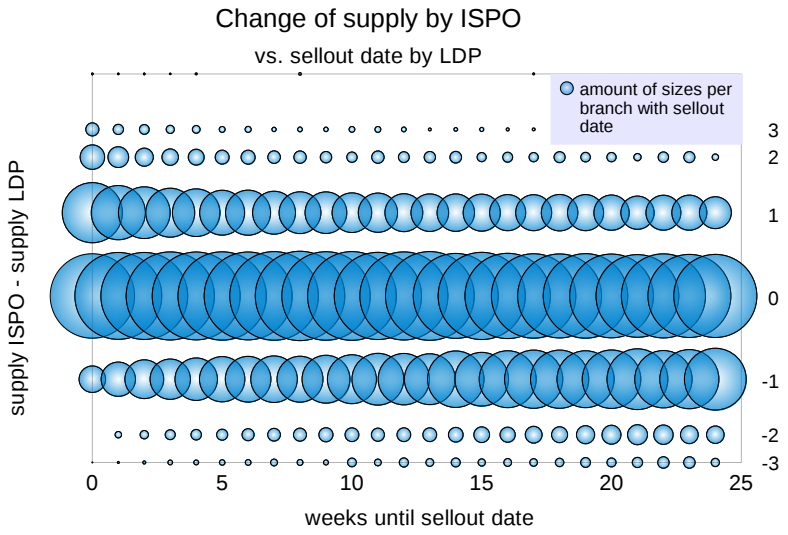

Figure 10.2: Change of supply by ISPO

10.4 Potential of ISPO

By integrating price optimization in the decision on the supply we hope to increase the realized revenue.

To get an idea of how good ISPO meets the realized demand of the different sizes we performed a test on 136 real instances for which we were in possession of transaction data. Because at this time all articles at our industrial partner were supplied by the LDP we can use real sales figures to compare the two models LDP and ISPO.

In Figure 10.2 we depicted the change of supply by ISPO against the supply that LDP yields. The size of the bubbles is related to the summed up number of sizes over all branches and articles which are sold out in the week marked at the x-axis. The bigger the bubble the more sizes per branch and article are sold out in that week.

The position of the bubbles on the y-axis describes the difference between the supply determined by ISPO and the realized supply computed by the LDP.

For example, let us consider an article for which a particular size in a particular branch is sold out 5 weeks after sales start. If for this article, branch and size ISPO would yield a supply of 5 and LDP would yield a supply of 3, then this article,branch and size would increase the relative size of the bubble at (5,2).

We see, that the size of bubbles decreases in terms of the sellout date for positive values on the y-axis, while the size increases for negative ones. This means, ISPO would deliver more items per branch and size the earlier the supplied items by the LDP were sold out and ISPO would deliver less items per branch and size the later the supplied items by the LDP were sold out. This is exactly the behavior we would expect to obtain a more size conform supply and an indication that ISPO leads to a more size conform supply than the LDP.

To measure the possible improvement also in terms of money we compared the

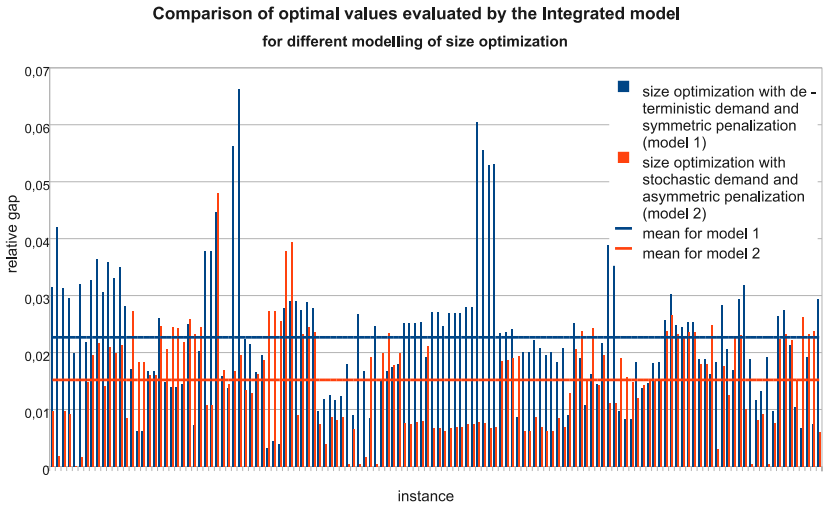

Figure 10.3: Comparision of different models for size optimization

three models LDP, SLPD and ISPO. For the same 136 instances we solved the LDP and SLDP exactly, ISPO – because of the long runtime – heuristically. To compare the three models with respect to monetary effects we evaluated them on the objective of ISPO.

For the optimal solutions of LDP and SLDP we computed the associated acquisition costs, pick costs and lot opening costs and then applied price optimization (Algorithm 3) to get the yield for the optimal price trajectory in terms of the supply. Subtracting all costs for supply from the yield provides the related objective value of ISPO.

The gaps between the objective of the heuristic solution of ISPO and the objective in terms of ISPO by fixing the supply computed by the LDP and SLDP are depicted in Figure 10.3. These gaps can be seen as a practical application of the value of the stochastic solution VSS, see Section 5.3.2. We measure the loss by non-regarding the second stage in the sales process or particularly in terms of the LDP by non-regarding random.

The vertical lines illustrate the relative improvement in terms of money for the particular instance. In terms of the LDP, ISPO could lead to an improvement about 2.27 percentage points of revenue, in terms of the SLDP about 1.52 percentage points.

The theoretical potential of increasing the revenue by 2.27 percentage points was encouraging and therefore our partner and the DISPO-team decided to deploy DISPO – ISPO combined with POP-RH – in a real-world field study.

10.5 DISPO – the field study

Parts of this section are presented in similar fashion in [KKR11b]. We performed our real-world field study as described above as a controlled statistical experiment to compare DISPO with the currently applied LDP together with manual price optimization.[2]

10.5.1 Preparation

Additionally to the estimation of the potential of ISPO we outlined in Section 10.4, there were some other work we/the DISPO-team and our partner were concerned with in the time before the field study began.

The DISPO-team had to confer with the partner about the exact test set-up. The in Section 10.2 described partition in test and control branches was chosen. To supply the test branches according to our proposal pre-packs had to be opened and items had to be removed or added by our industrial partner. Therefore the partner decided to use its German online-shop to compose the lots for the test branches after the lots arrived at the headquarters.

Moreover the participating commodity groups were chosen. These were three commodity groups with comparatively many sizes, see 10.5.2. The involved persons also arranged an appropriate point in time for the field study .

We met with those responsible of the sales department to come to an realistic estimation of the fixed and variable mark-down cost and the salvage value.

It was also discussed if advertising campaigns and special offers should be allowed. The results will follow in Subsection 10.5.2.

For performing POP-RH we could take the most scripts and programs developed by the former DISPO-team that were used in former field studies.

To detect potential weak points and to practice the procedure on both sides, we performed a test field study together with the IT department of our partner. Additionally potential inconsistencies in terms of the used data formats should be eliminated. The test data our partner provided us included relevant data for all planned orders for the last season of the year 2010, historical data for demand estimation and current transaction data for all commodity groups. Overall this was about 6.6 GB of data.

For this data – after we performed the empirical estimation – we computed a supply policy by ISPO-PingPong and provided our partner the results for the test branches.

We also tested the process for performing POP-RH. That means with respect to latest sales figures we computed an optimal mark-down strategy via the $\text{POP}^{\hat{e}}$ and informed our partner about proposed mark-downs for the subsequent two weeks.

The test field study among others led to some adjustments of scripts and readin routines.

10.5.2 Setup of the field study

It was necessary to select a small set of articles for the field study because the orders had already been placed in terms of lot-types, and the adaption of the supply for the test branches to the results of the new method is a too expensive logistic operation to be carried out for each article.

[2] A comparision between ISPO and complete manual planning was not possible, because nearly all articles at our industrial partner are supplied by the LDP now.

The test branches were supplied in terms of the – by ISPO-PingPong (Algorithm 15) computed – solution of ISPO. Since there are global constraints for the overall number of supplied items we actually *computed* the supply for *all* branches with the new method – and so did our industrial partner with the LDP. Our proposed supply was then implemented only for the test branches; the supply of the control branches (and also all remaining branches) was implemented as computed by the LDP by our project partner.

The field study ran from end of May until end of September 2011 for 81 articles from three different commodity groups – women overgarments fashion (wof), women overgarments classic (woc) and women underwear (wu). The sales process for these articles started between May 2011 and mid of June 2011 so that all articles could be observed for a time period of 15 to 17 weeks. Some further relevant properties of the used test articles are stated in Table 10.2. Our demand estimation, see 3.2, is based on historical data in a time frame from September 2009 to September 2010.

commodity group	number of articles	number of sizes
wof	9	6
woc	9	3
wu	5	6

Table 10.2: Properties of the test articles.

Table 10.3 shows the parameter setting we used in ISPO for the field study. Our lot-type opening costs δ_i and pick costs pcost were estimated on the basis of a thourough cost accounting. This cost accounting also revealed that more than four lot-types can only be handled if the area for internal stock-turnover is increased substantially. The discount factor ρ is derived from an estimation of the capital binding cost. Whenever other reasons than interest rates favor faster stock-outs this can be increased. The fact that we did not account for mark-down costs μ_k just reflects the fact that at the time of the design of the experiment our partner could simply not provide a realistic value for this. An adaption of ISPO-PingPong was not possible until the beginning of the experiment due to time constraints.

parameter	setting
κ	4
pcost	0.0545
δ_1	100
$\delta_i, i > 1$	50
E	{low, normal, high} (period-0 sales $\in [0, 10\,\%)/[10\,\%, 30\,\%]/(30\,\%, 100\,\%]$)
$d^{\text{normal}}_{k,p,b,s}$	from empirical distribution and interpolation of historic sales in commodity group
$d^{e}_{k,p,b,s}$	$\alpha \times d^{\text{normal}}_{k,p,b,s}$ (α from historic sales in scenario e compared to scenario normal)
$\mathrm{Prob}(e)$	from empirical distribution of historic sales in commodity group
k_{obs}	2 (i.e., realization of e and earliest mark-down after 2 periods)
$k_{\max}$	periods (= weeks) until end of season (article dependent)
ρ	0.000974868
$p_{\max}$	4 (five prices including start price and salvage value)
μ_k	0
$\pi_{k_{\max}}$	dependent on commodity group $\in [15\%, 30\%]$ of the starting price π_0

Table 10.3: Parameter setting for the field study.

Advertising campaigns and special offers

From time to time our industrial partner performs different campaigns – triggered by exogeneous reasons like new competing stores and the like – in its branches. Articles from a commodity group are all marked down at the same time or they sell three shirts for the price of two and so on.

The question was how to deal with this? Prohibiting these measures would certainly make the result less biased. However, if our partner decided to implement DISPO this effects would occur anyway. DISPO has to deal with them. In order to better assess the practicability of DISPO, we decided not to forbid campaigns in the test and control branches: A method the performance of which vitally relies on laboratory conditions with all exogenous disturbances removed cannot be used in practice anyway.

Stock transfers

There are two types of stock transfers. On the one hand branches perform stock transfers caused by single customer requirements. If the requested product is not available in a branch the employees have the possibility to obtain it from an other branch. On the other hand there are systematical stock transfers which are caused by an undersupply of a size in a branch. Then the sales department decides to restock items from a oversupplied branch to a branch where there is a shortage.

While the transfers caused by customer requirements in our partner's view for the field study can not be forbidden the DISPO-team and our partner decided to prohibit all systematical stock transfers. Because they compensate wrong supply – maximizing the realized revenue by a size-compliant supply is the aim of our method – they would bias the result in such a way that it might be useless.

10.5.3 Evaluation

Our test set of articles is denoted by A. For reasons of comparability we consider for each branch the objective value of ISPO divided per merchandise value over all articles from the set A. We distinguish the corresponding variables and parameters for the different articles $a \in A$ by a superscript a. Apart from that the parameters name are identical to the formulation of ISPO, Problem 6.

For an initial stock $I^a_{b,s}$ for the considered branch b, size s and and a starting price π^a_0 we compute the *relative realized objective for branch* b of the independent non-anticipative decisions

$$RRO_{\mathrm{ISPO}}(b) = \frac{\text{objective of ISPO achieved for } b}{\text{maximal possible objective for } b} = \frac{-\sum_{a\in\mathcal{A}}\sum_{\ell\in L}\sum_{m\in M} x^a_{b,\ell,m}\cdot c^a_{b,\ell,m} - \sum_{i=1}^{\kappa}\tilde{\delta}_i\cdot z^a_i + \sum_{k\in K}\exp(-\rho k)\Big(\sum_{a\in\mathcal{A}}\sum_{s\in S}\hat{r}^a_{k,b,s} - \tilde{\mu}^a_k\hat{n}^a_k\Big)}{-\sum_{a\in\mathcal{A}}\sum_{\ell\in L}\sum_{m\in M} x^a_{b,\ell,m}\cdot c^a_{b,\ell,m} - \sum_{i=1}^{\kappa}\tilde{\delta}_i\cdot z^a_i + \sum_{a\in\mathcal{A}}\sum_{s\in S} I^a_{b,s}\cdot\pi^a_0}. \quad (10.7)$$

Depending on article a, the entity z^a_i indicates that an i-th lot-type was used. During the sales process, we observed $\hat{r}^a_{k,b,s}$ (the realized yield for Branch b and Size s in Period k) for Article a and $\hat{n}^a_k$ (mark-down in Period k – yes or no).

Since we only consider a subset of branches we have to take into account that pick costs, costs for additional lot types, and fixed mark-down costs must be scaled with respect to the number of considered branches. This way, we get a marginal cost $\tilde{\delta}_i$

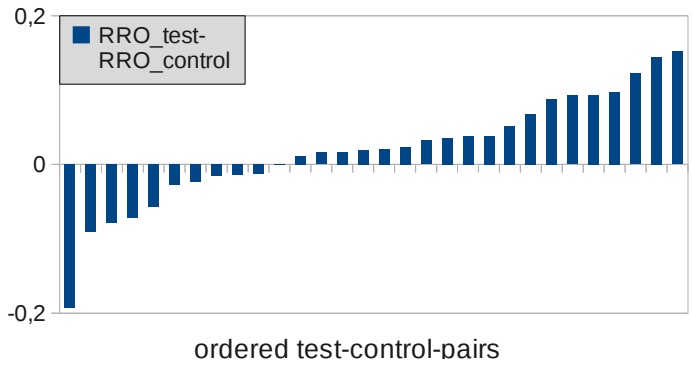

Figure 10.4: $RRO_{\text{test}} - RRO_{\text{control}}$ for ordered test-control-pairs – all 81 articles

for the i-th selected lot-type and mark-down costs $\tilde{\mu}_k^a$ for period k. (For a complete notational reference see the problem formulation of ISPO in Chapter 6).

For the evaluation of the field study we used transaction data from our industrial partner, analogous to the historical data we use for demand estimation. From this data we obtained the daily sales per branch and size and the realized sales price. But this data yields not all information we needed. We did not have full access to the supply in terms of lot-types in the control branches because the partner did not store the data for all articles. So we had to reconstruct the lot-types from the transaction data. In the most cases we can exactly determine the lot-type which is used in a branch. But we know from former field studies that not for all branches exactly the computed lot-type is also delivered. Sometimes there is one item less ore more supplied for a size. We therefore counted a lot-type as ”new“ lot-type if at least three branches were supplied with this lot-type. Otherwise we assumed that this lot-type resulted from an ”old“ lot-type by wrong delivery. Then for this ”wrong“ lot-type only one time pick costs were counted. We want to emphasize that this kind of adaption – if it affects the results – is a disadvantage for the test branches and thus for DISPO. Because every branch has to be supplied by at least one lot-type pick cost always arise at least once.

Allowing campaigns and special offers sometimes leads to the fact that the realized sales price differs from the determined price – either in the test branches or in the control branches. Moreover we cannot exclude that some items are bought at a reduced rate because of material defect or the like. There are two possibilities: Either we take the determined price per week for the computation of the RRO or the realized one. We decided for the realized price. We want to test the real-world behavior of DISPO and as already stated in 10.5.2 mark-downs beyond the determined prices are part of it.

10.5.4 Results of the field study

The relative realized revenues per branch are shown in Table 10.4 for each test-control pair in the second and third column.

We see that on average, the use of the new method gains almost two percentage points compared to the old method.

We apply the Wilcoxon signed-rank test. The differences of the observations, here $RRO_{\text{test}} - RRO_{\text{control}}$ – at the fourth column of Table 10.4 are ordered increasingly according to their absolute values (depicted in Figure 10.4). The signed ranks are

test-control-pair	RRO_{test}	RRO_{control}	$RRO_{\text{test}} - RRO_{\text{control}}$	signed rank
1	0.6333	0.6214	0.0119	2
2	0.6764	0.6080	0.0683	19
3	0.5919	0.6072	−0.0154	−5
4	0.6056	0.5898	0.0159	6
5	0.6637	0.5663	0.0974	26
6	0.6228	0.6031	0.0197	8
7	0.6377	0.6500	−0.0123	−3
8	0.5832	0.5845	−0.0013	−1
9	0.5968	0.5731	0.0237	11
10	0.5372	0.6276	−0.0904	−23
11	0.5651	0.5489	0.0163	7
12	0.5333	0.5904	−0.0571	−18
13	0.5782	0.5570	0.0212	9
14	0.6381	0.4940	0.1441	28
15	0.5054	0.5845	−0.0791	−21
16	0.5927	0.4993	0.0934	25
17	0.5872	0.4943	0.0929	24
18	0.6078	0.5691	0.0388	16
19	0.5762	0.6476	−0.0714	−20
20	0.5682	0.5323	0.0359	14
21	0.5133	0.4250	0.0883	22
22	0.5272	0.5547	−0.0275	−12
23	0.4015	0.5942	−0.1926	−30
24	0.4628	0.4860	−0.0232	−10
25	0.5168	0.4646	0.0522	17
26	0.5843	0.4621	0.1222	27
27	0.5658	0.4137	0.1521	29
28	0.4989	0.4608	0.0380	15
29	0.5466	0.5607	−0.0141	−4
30	0.5593	0.5272	0.0320	13
∅	0.5692	0.5499	0.0193	5.7

Table 10.4: RROs for the test-control-pairs – all 81 articles

stated in the fifth column of the table. At first glance we can see that the differences are not distributed equally. The test branches perform visibly better. This is also the result of the Wilcoxon signed-rank test. For the data we get a rank-sum of 318. The probability for getting an equal or higher rank-sum is $P_{30}(X \geq 318) \approx 4.02\%$. Thus, with a probability of 4.02% for a better performance of the test branches resulted by chance we get for the 81 test articles a significant result for an improvement of DISPO – against the LDP with manual planning of mark-downs.

However, we observed that some operational anomalies like failed price cuts in the control branches. In order to estimate the influence of the new method in the most conservative fashion, we removed all articles which may have been affected by systematic disturbances of operations. This led to a second set of articles A' with only 23 articles remaining.

The particular RROs per branch are stated in Table 10.5, the corresponding differences $RRO_{\text{test}} - RRO_{\text{control}}$ are depicted in Figure 10.5. We see that in the case of heavily cleaned-up data the RRO for the test branches is still more than 1.5 percentage points higher than in the control branches. We repeated the Wilcoxon signed-rank test for this smaller test set. The test now yields a rank-sum of 271, which leads to a probability of $P_{30}(X \geq 271) = 22\%$ that a better performance of the test-branches was observed by chance. Thus, for the heavily cleaned-up data we still observe a relevant effect (1.5 percentage points improvement) whose observation can no longer be testified as significant. This is essentially caused by the fact that for such a small (but relevant) effect the sample set A' is simply no longer large enough to prove significance. Still, the probability for a randomly better performance is with 22% much higher than the

test-control-pair	RRO_{test}	RRO_{control}	$RRO_{\text{test}} - RRO_{\text{control}}$	signed rank
1	0.4215	0.6673	−0.2458	−28
2	0.5874	0.4758	0.1116	18
3	0.6572	0.4865	0.1708	25
4	0.5948	0.4773	0.1175	21
5	0.5491	0.4153	0.1338	24
6	0.5799	0.5117	0.0682	13
7	0.4833	0.5454	−0.0621	−12
8	0.4648	0.5124	−0.0476	−9
9	0.5051	0.4923	0.0128	2
10	0.4933	0.6094	−0.1162	−19
11	0.4926	0.4998	−0.0071	−1
12	0.4205	0.4706	−0.0501	−10
13	0.4352	0.3746	0.0607	11
14	0.7046	0.2860	0.4186	30
15	0.4547	0.5281	−0.0734	−14
16	0.5146	0.3846	0.1300	22
17	0.5285	0.4247	0.1038	17
18	0.4802	0.5081	−0.0279	−3
19	0.3562	0.4865	−0.1303	−23
20	0.4119	0.4496	−0.0377	−5
21	0.2195	0.2577	−0.0382	−6
22	0.4274	0.5437	−0.1163	−20
23	0.2262	0.6415	−0.4153	−29
24	0.4006	0.3252	0.0754	16
25	0.3779	0.4244	−0.0465	−8
26	0.4759	0.4008	0.0750	15
27	0.5926	0.3971	0.1955	26
28	0.4458	0.4116	0.0342	4
29	0.4540	0.4985	−0.0445	−7
30	0.5278	0.3050	0.2228	27
∅	0.4761	0.4604	0.0157	2.57

Table 10.5: RROs for the test-control-pairs – heavily cleaned-up data, 23 articles

sample	relative realized objective	gross yield	sales
test	0.4761	0.6829	0.7951
control	0.4604	0.6744	0.8021

Table 10.6: Alternative performance metrics, heavily cleaned-up data.

probability for a randomly worse performance given by $P_{30}(X \leq 271) = 78.6\%$.

So far, we assessed the quality of the decisions of the various methods on the basis of our objective function that was carefully engineered together with our partner. Yet, it is interesting to see that the new two-stage method outperforms the old method in some very important criteria at the same time. In Table 10.5.4 we list average RRO per branch, relative gross yields, and relative sales for all test-control-pairs. For both revenue and gross yield we see improvements. In contrast to this, the number of sales is only minimally smaller for DISPO.

Now, which decisions have been taken differently by the new method? Table 10.7 shows the differences in the lot-type designs of the new and the old method for the 23 remaining articles.[3] The most obvious effect is that the number of different lot-types used is usually smaller for the ISPO than for the LDP. Since the old method tries to approximate a fractional demand as closely as possible by a supply distribution on

[3]Since the lot-type design of the control branches had to be reconstructed from incomplete data – see Subsection 10.5.3, the multiplicities for the control branches do not always add up to 30. The lot-types are reliable, though.

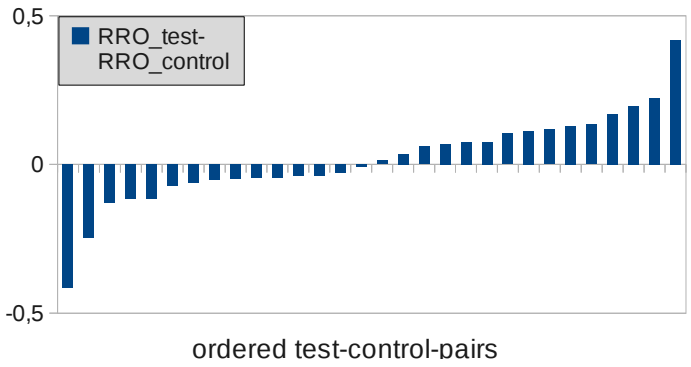

Figure 10.5: $RRO_{\text{test}} - RRO_{\text{control}}$ for ordered test-control-pairs – heavily cleaned up data, 23 articles

the basis of suitable lot-types, it will usually use as many lot-types as possible, even if the improvements of a new lot-type are small. The goal of the new method is not to meet the demand as closely as possible but to earn as much money as possible. Obviously, an additional lot-type is not always justified by higher predicted profits in ISPO. Consequently, ISPO does not suggest to use such a new lot-type. In the table we clearly see that lot-type $(1, \ldots, 1)$ is very often used. This is the result of the rule that each branch has to receive at least one piece in every size – a fact that reduces the potential for improvement and should be taken into account when the effect (1.5 to 2 percentage points improvement) of using the new method is assessed.

no.	lots delivered to test branches by new method	lots delivered to control branches by old method
1	4(2,2,3,4,3,3),19(1,1,1,1,1,1),7(1,1,2,2,2,2)	13(1,1,1,1,1,1),7(1,1,1,2,2,1),5(1,1,2,2,3,2),3(2,3,3,4,4,3)
2	4(2,2,3,4,3,3),19(1,1,1,1,1,1),7(1,1,2,2,2,2)	13(1,1,1,1,1,1),7(1,1,1,2,2,1),5(1,1,2,2,3,2),3(2,3,3,4,4,3)
3	5(2,2,3,4,3,3),18 (1,1,1,1,1,1),7(1,1,2,2,2,2)	15(1,1,1,1,1,1),7(1,1,1,2,2,1),3(2,3,3,4,4,3),3(1,1,2,2,3,2)
4	5(2,2,3,4,3,3),18(1,1,1,1,1,1),7(1,1,2,2,2,2)	12(1,1,1,1,1,1),8(1,1,1,2,2,1),3(1,1,2,2,3,2),3(2,3,3,4,4,3)
5	5(2,2,3,4,3,3),18(1,1,1,1,1,1),7(1,1,2,2,2,2)	13(1,1,1,1,1,1),7(1,1,1,2,2,1),4(1,1,2,2,3,2),4(2,3,3,4,4,3)
6	5(2,2,3,4,3,3),18(1,1,1,1,1,1),7(1,1,2,2,2,2)	14(1,1,1,1,1,1),7(1,1,1,2,2,1),6(1,1,2,2,3,2),3(2,3,3,4,4,3)
7	5(2,2,3,4,3,3),18(1,1,1,1,1,1),7(1,1,2,2,2,2)	14(1,1,1,1,1,1),7(1,1,1,2,2,1),6(1,1,2,2,3,2),3(2,3,3,4,4,3)
8	5(2,2,3,4,3,3),18(1,1,1,1,1,1),7(1,1,2,2,2,2)	12(1,1,1,1,1,1),7(1,1,1,2,2,1),7(1,1,2,2,3,2),3(2,3,3,4,4,3)
10	13(1,1,1,2,2,2),17(1,1,1,1,1,1)	6(2,2,2,3,4,4),8(1,1,1,2,3,3),12(1,1,1,2,2,2),4(1,1,1,1,1,1)
11	13(1,1,1,2,2,2),17(1,1,1,1,1,1)	6(2,2,2,3,4,4),8(1,1,1,2,3,3),12(1,1,1,2,2,2),4(1,1,1,1,1,1)
12	13(1,1,1,2,2,2),17(1,1,1,1,1,1)	6(2,2,2,3,4,4),8(1,1,1,2,3,3),12(1,1,1,2,2,2),4(1,1,1,1,1,1)
14	13(1,1,1,2,2,2),17(1,1,1,1,1,1)	9(1,1,2,2,2,2),9(1,1,1,1,1,1),3(1,1,2,2,1,1),6(1,1,1,1,2,2)
16	10(1,1,1,2,2,2),7(1,1,2,2,2,2),13(1,1,1,1,1,1)	14(1,1,2,2,3,3),5(2,2,3,4,4,4),11(1,1,1,2,2,2)
17	10(1,1,1,2,2,2),7(1,1,2,2,2,2),13(1,1,1,1,1,1)	14(1,1,2,2,3,3),5(2,2,3,4,4,4),11(1,1,1,2,2,2)
18	10(1,1,1,2,2,2),7(1,1,2,2,2,2),13(1,1,1,1,1,1)	14(1,1,2,2,3,3),5(2,2,3,4,4,4),11(1,1,1,2,2,2)
19	18(3,2,1),12(2,1,1)	10(4,2,1),19(3,2,1)
20	8(1,3,2),22(1,2,1)	10(1,2,1),6(2,4,3),11(1,3,2),2(1,1,1)
21	8(1,3,2),22(1,2,1)	22(1,2,1),6(1,1,1),2(1,3,1)
22	7(2,4,3),11(1,2,1),4(2,3,2),8(1,3,2)	16(1,2,1),7(2,4,3),3(1,3,2),1(1,2,2)
23	18(3,2,1),12(2,1,1)	1(2,1,1),9(4,2,1),18(3,2,1),1(1,1,1)

Table 10.7: Supply for the test and control branches in terms of lots.

article	RRO		#mark-downs	
article	test	control	test	control
1	0.5137	0.5171	1	1
2	0.5177	0.4939	1	1
3	0.6578	0.5956	1	1
4	0.5738	0.6391	1	0
5	0.6494	0.6620	1	1
6	0.6477	0.6012	0	1
7	0.5314	0.4452	0	1
8	0.5090	0.3518	1	1
9	0.2493	0.1840	1	1
10	0.2363	0.1848	1	1
11	0.4246	0.4167	1	1
12	0.6183	0.6079	1	1
13	0.2924	0.1982	0	1
14	0.6257	0.6462	0	1
15	0.7812	0.7817	0	1
16	0.7812	0.7295	0	1
17	0.3164	0.3175	1	1
18	0.2864	0.2184	1	1
19	0.6225	0.6358	0	0
20	0.5017	0.5591	0	0
21	0.5553	0.5592	0	1
22	0.6240	0.6229	1	0
23	0.4824	0.5216	1	0
∅	0.5217	0.4995	0.6087	0.7826

Table 10.8: RROs and mark-downs per article

Also in terms of performing mark-downs differences are recognizable. On the heaviliy cleaned-up data set of 23 articles, the new price optimization suggested alltogether 14 mark-downs in the test branches, while the manual strategy in the control branches led to 18 mark-downs on the same set. To evaluate the influence of mark-downs we consider the *realized relative revenue for article* a. It is given by

$$RRO_{\mathrm{ISPO}}(a) = \frac{\text{objective of ISPO achieved for } a}{\text{maximal possible objective for } a} = \frac{-\sum_{b\in B}\sum_{\ell\in L}\sum_{m\in M} x^a_{b,\ell,m}\cdot c^a_{b,\ell,m} - \sum_{i=1}^{\kappa}\tilde{\delta}_i\cdot z^a_i + \sum_{k\in K}\exp(-\rho k)\Big(\sum_{b\in B}\sum_{s\in S}\hat{r}^a_{k,b,s} - \tilde{\mu}^a_k\hat{n}^a_k\Big)}{-\sum_{b\in B}\sum_{\ell\in L}\sum_{m\in M} x^a_{b,\ell,m}\cdot c^a_{b,\ell,m} - \sum_{i=1}^{\kappa}\tilde{\delta}_i\cdot z^a_i + \sum_{b\in B}\sum_{s\in S} I^a_{b,s}\cdot\pi^a_0}. \quad (10.8)$$

See the formulation of ISPO, Problem 6 for the exact notation.

We state the RROs per article in the set A' for the test-control branches in Table 10.8. Moreover the number of mark-downs per article for the test and for the control branches are stated in the third and fourth column.

In the test branches the mean relative revenue per article amounts to 2.22 percentage points more than in the control branches. To see in which cases we performed better than the sales department with its manual planning we divided the articles in four subsets. We consider

1. articles which are marked down as well in the test-branches as in the control branches – also denoted as ”both“,
2. articles which are marked down only in the test-branches – ”just test“,
3. articles which are marked down only in the control-branches – ”just control“,
4. articles which are marked down neither in the test nor in the control branches – ”none“.

	both		just test		just control		none	
	test	control	test	control	test	control	test	control
RRO	0.4526	0.4136	0.5601	0.5945	0.6021	0.5659	0.5621	0.5974
sample size	11		3		7		2	

Table 10.9: Mean RROs versus mark-downs

	com. group	no.	predicted	realized	gap
	wof	1	527.96	318.64	−0.3965
	wof	2	527.96	285.88	−0.4585
	wof	3	900.39	490.29	−0.4555
	wof	4	900.39	603.34	−0.3299
	wof	5	900.39	533.66	−0.4073
	wof	6	391.20	482.88	0.2343
	wof	7	391.20	415.96	0.0633
	wof	8	700.42	656.63	−0.0625
	wof	9	700.42	521.56	−0.2554
	woc	10	620.24	440.91	−0.2891
	woc	11	620.24	435.63	−0.2976
	woc	12	620.24	497.15	−0.1984
	woc	13	957.87	666.63	−0.3041
	woc	14	957.87	545.82	−0.4302
	woc	15	957.87	622.68	−0.3499
	woc	16	631.84	680.98	0.0778
	woc	17	631.84	664.36	0.0515
	woc	18	631.84	651.06	0.0304
	wu	19	393.90	292.29	−0.2580
	wu	20	624.91	414.50	−0.3367
	wu	21	262.52	297.26	0.1324
	wu	22	371.35	508.94	0.3705
	wu	23	421.82	364.31	−0.1363
∅					−0.1742
sd					0.2388

Table 10.10: Comparison of objective function values – predicted by ISPO versus realized.

The mean RROs per article according to these subsets of articles are stated in Table 10.9. In the case that the article is marked down as well in the test branches as in the control branches we can observe an about 3.9 percentage points higher revenue for the test branches. In the case that the articles were just marked down in the control branches this difference amounts to 3.62 percentage points. For the set of articles where only mark-downs were performed in the test branches the control branches yield averagely 3.44 percentage points higher revenue and 3.53 percentage points in the subset of articles for which in both samples no mark-downs are performed: We perform better solely in the cases where mark-downs were performed for the control branches.

Yet, the sample sizes are very small. So it is not possible to assess significant results.

Nevertheless, let us consider the case were mark-downs are performed in both samples. What are the differences in the mark-down decisions of DISPO and the mark-downs decided by our partner? DISPO as a rule decides later on mark-downs. In seven of the eleven articles ISPO proposed a later mark-down – averagely 10 days – than the sales department at our partner. For these articles an about 2.69 percentage points higher RRO can be observed. If DISPO decides on an earlier mark-down than our partner it is averagely 33 days earlier. For these articles our method yields averagely an about 6.02 percentage points higher RRO.

	com. group	no.	predicted	realized	gap
	wof	1	232.295	194	−0.1649
	wof	2	232.295	177	−0.2380
	wof	3	231.724	180	−0.2232
	wof	4	231.724	210	−0.0937
	wof	5	231.724	198	−0.1455
	wof	6	227.202	235	0.0343
	wof	7	227.202	214	−0.0581
	wof	8	225.305	253	0.1229
	wof	9	225.305	226	0.0031
	woc	10	204.52	203	−0.0074
	woc	11	204.52	206	0.0072
	woc	12	204.52	207	0.0121
	woc	13	199.606	221	0.1072
	woc	14	199.606	204	0.0220
	woc	15	199.606	218	0.0922
	woc	16	225.639	251	0.1124
	woc	17	225.639	237	0.0504
	woc	18	225.639	235	0.0415
	wu	19	138.836	125	−0.0997
	wu	20	122.689	95	−0.2257
	wu	21	124.373	123	−0.0110
	wu	22	191.482	213	0.1124
	wu	23	138.836	130	−0.0636
∅					−0.0267
sd					0.1133

Table 10.11: Comparison of sales – predicted by ISPO versus realized.

In Tables 10.10 and 10.11 we show how well ISPO predicted the expected function values and the expected sales. While the prediction quality of the expected function values seems unsatisfactory, the prediction of sales is quite good. That sales can be predicted well is more an indication for the fact that essentially everything is sold anyway. What matters more is how much money can be earned by these sales. And this in turn indicates that it is vital to estimate the return when it comes to decide about the distribution of supply. Although our predictions are presumably biased, the volatility even in one commodity group is very high (expressed by the standard deviation): a gap of zero is still inside the interval “average minus standard deviation” through “average plus standard deviation”.

Chapter 11

Conclusion

Our aim was to develop a decision support system for the optimization of supply at a fashion retailer.

In this context we deployed the two-stage stochastic program ISPO where the first stage is the supply in terms of lot-types (size optimization) and mark-downs act as recourse (price optimization).

To exploit current sales figures, after the by ISPO computed supply is adopted by the industrial partner, the mark-down strategy is updated every week. Therefore we use a closed looped policy developed by the former DISPO-team. For practical purposes we had to develop the faster approach POP-DYN for solving the underlying Price Optimization Problem. We devised dominance rules for mark-down strategies and exploited them in a fast dynamic programming approach.

We proposed the Branch&Bound solver ISPO-BAB to compute an exact solution of ISPO. The principal idea is to enumerate all price-trajectories for each scenario a priori and then to solve the size optimization stage for the fixed strategies. By mapping scenarios to price trajectories ISPO simplifies to an SLDP which can be solved by state-of-the-art MIP solvers. To prune the Branch&Bound tree we apply dual bounds based on the wait-and-see solution from stochastic programming. We extended the wait-and-see solution by considering subsets of scenarios. To accelerate our algorithm we relaxed the wait-and-see solution, on the one side by permitting independent single supply for branches and sizes instead of lot-types and on the other side by LP relaxations. We compute costlier dual bounds only on demand.

Still, our exact solver only serves us as a benchmarking tool. For practical purposes we developed the fast heuristic ISPO-PingPong with a small optimality gap. This approach exploits the fact – we call it reversible recourse – that for every valid price trajectory there exists a feasible supply policy.

We performed a field study as a statistical experiment where we compared test against control branches. We were able to evidence averagely higher realized revenues of DISPO against a manual mark-down policy together with a supply policy – based on the LDP developed by the former DISPO-team – that does not regard the effect of mark-downs.

Both, ISPO-PingPong in terms of goodness and efficiency and the fast optimal dynamic programming solver POP-DYN for the weekly adaption of the mark-down strategy could be applied by our industrial partner

Until now an MIP solver is used to solve the underlying Stochastic Lot-type Design Problems in the heuristic ISPO-PingPong. We stated how to reduce an SLDP to κ

LDPs. The particular LDPs could fast be solved by applying the SFA heuristic that does not use any (commercial) state-of-the-art MIP solver. If our industrial partner decided to implement ISPO-PingPong our first step would be to adapt it in this way to economize license fees. We expect that this adaption would only have a small influence on the optimality gap of the solution.

In terms of the cooperation with our industrial partner our focus was mainly on the practical side. With the development of DISPO we succeeded. Moreover we could obtain some theoretical results as the dominance rules for the price optimization problem or our extended wait-and-see solution. A starting point for future research is the – theoretical and empirical – comparison of extended wait-and-see solutions with common dual bounds from stochastic programming like the introduced bounds derived from group subproblems. We are also interested in finding other two-stage stochastic problems with a similar structure, i.e. reversible recourse. We hope for the possibility to apply the ideas of our exact solver to them. It may be that ISPO-BAB could be generalized for these kind of problems.

We want conclude this thesis by adding the most important practical results that were obtained during the cooperation with our industrial partner to Figure 1.1 from the introduction.

In a preliminary study the former DISPO-team could evidence a mean improvement of 0.85 percentage points in terms of the gross yield by replacing former used standard lot-types with the supply that is proposed by the Lot-type Design Problem LDP. According to what our industrial partner says the actual improvement is much higher. In a field study in terms of Price optimization on its own we observed an about 0.76 percentage points higher relative realized revenue for the test branches. By integrating size and price optimization the improvement in terms of the relative realized revenue amounts to 1.57 percentage points: Supply and mark-down strategies interact substantially. The decision support system DISPO tackles this interaction and is suitable for practical use.

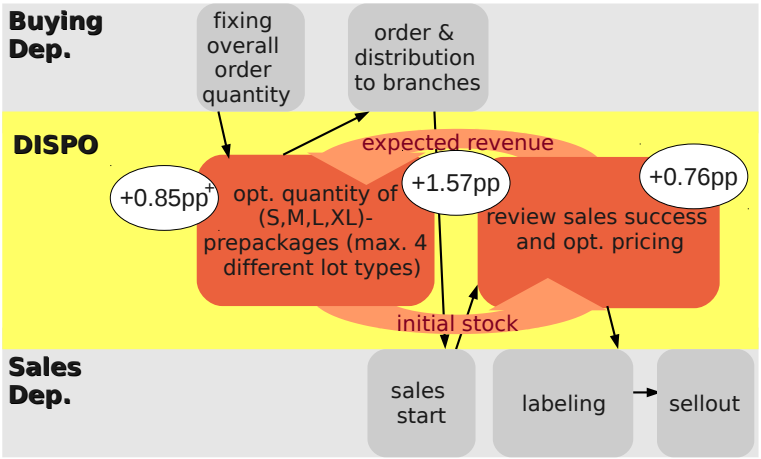

Figure 11.1: Integration of DISPO into the business process – results

Appendix A

ISPO-PingPong – further results

At this point we depict further computational results for ISPO-PingPong, Algorithm 15. For all test instances from the set $\mathcal{I}_6^{\text{test}}$, Appendix E, we depict optimality gaps per half iteration (size or price optimization stage), progress of optimality gaps, number of iterations and runtime. We consider maximal solving times tb of 20 and 60 seconds for the SLDP$_{\hat{L}}\left(W^E\right)$ with 50 or 100 traversed κ-subsets, nr^κ. The results for 20 seconds and 100 subsets are stated in Chapter 9.

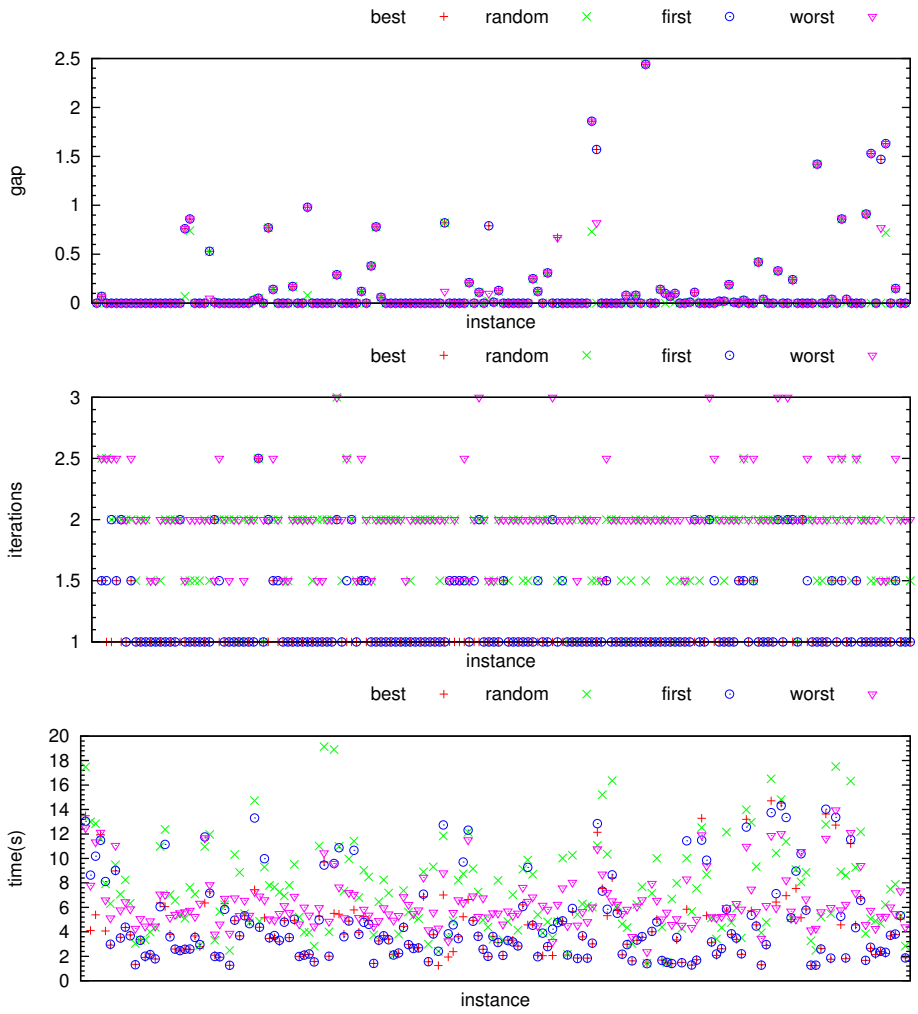

Figure A.1: 20 seconds solving time and 50 κ-subsets

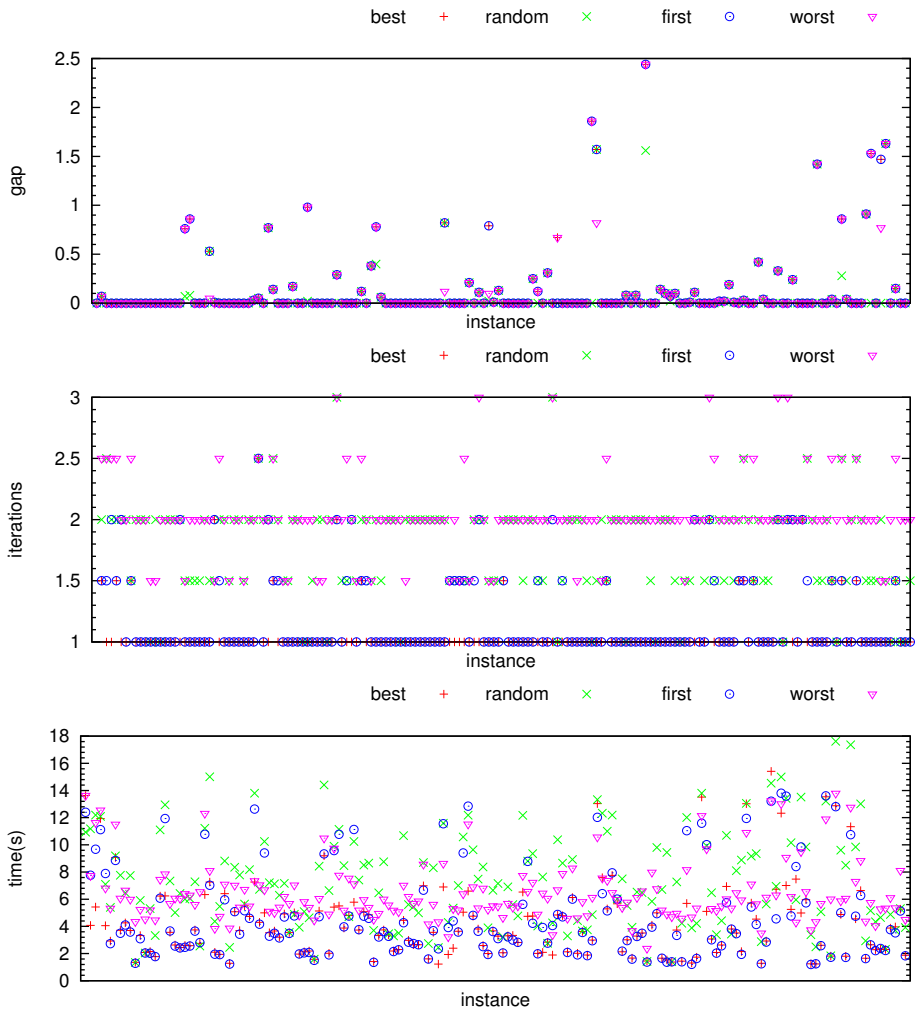

Figure A.2: 60 seconds solving time and 50 κ-subsets

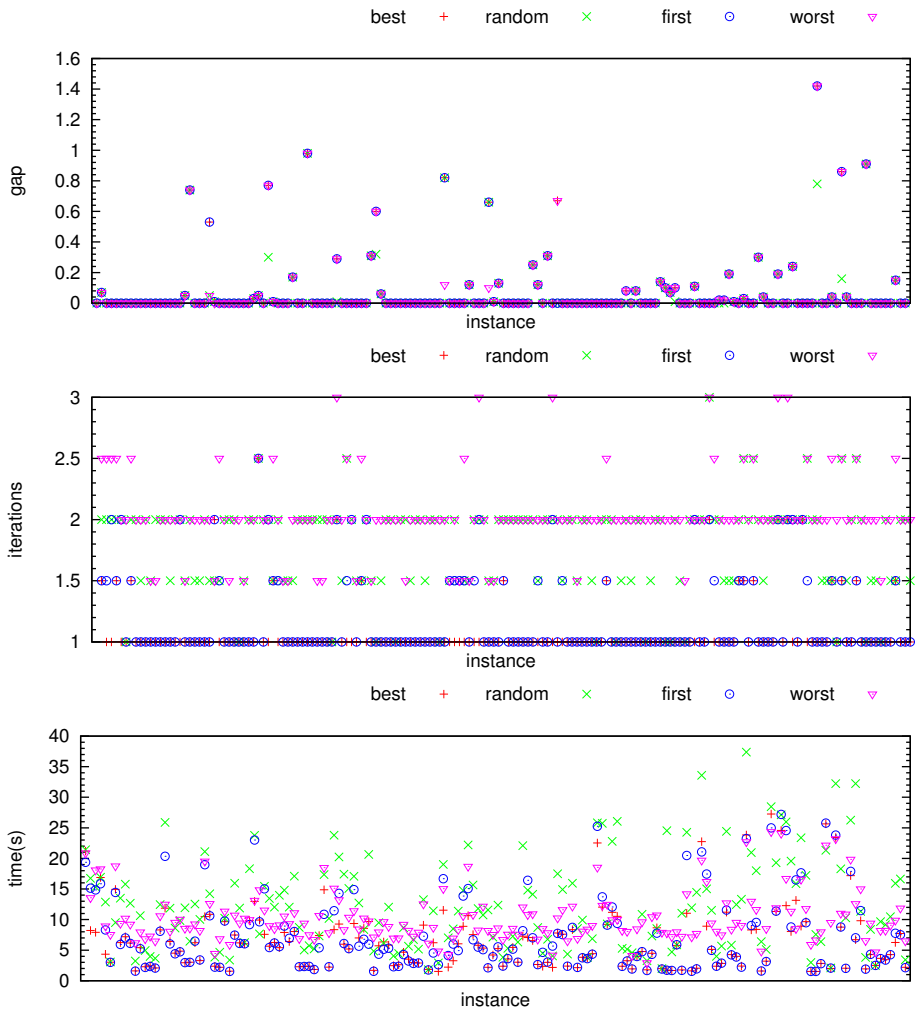

Figure A.3: 60 seconds solving time and 100 κ-subsets

Appendix B

Sales increase by mark-downs – further results

In Chapter 10, Figure 10.1, we illustrated the sales increase caused by mark-downs for a subset of articles in the related field study. At this point we present the results for the rest of the articles.

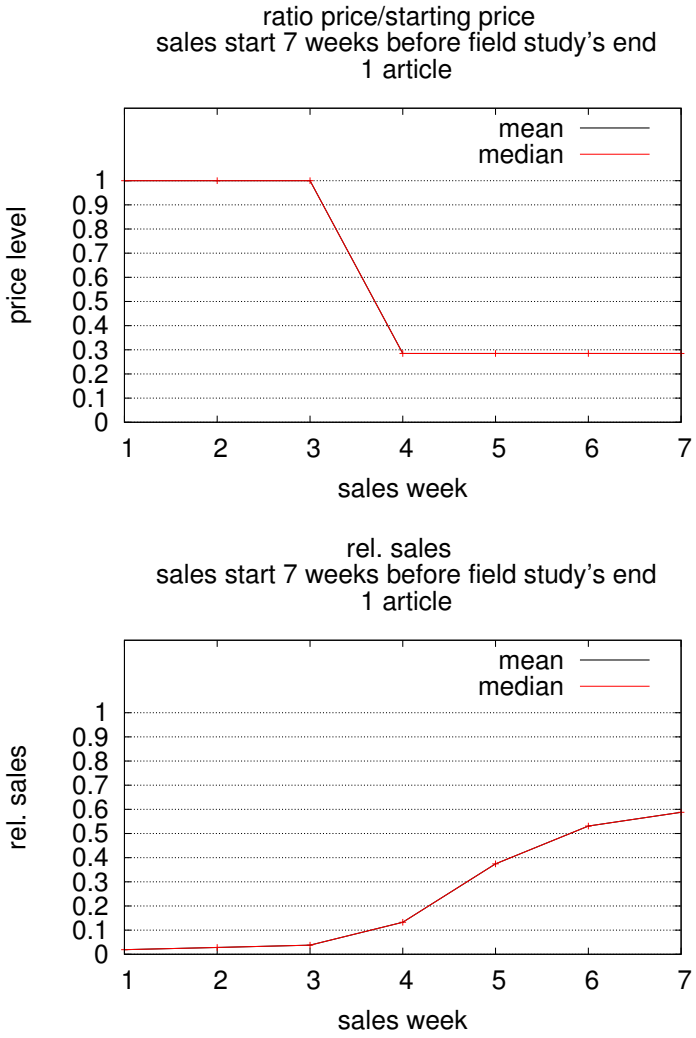

Figure B.1: Effect of mark-downs – 7 weeks selling time

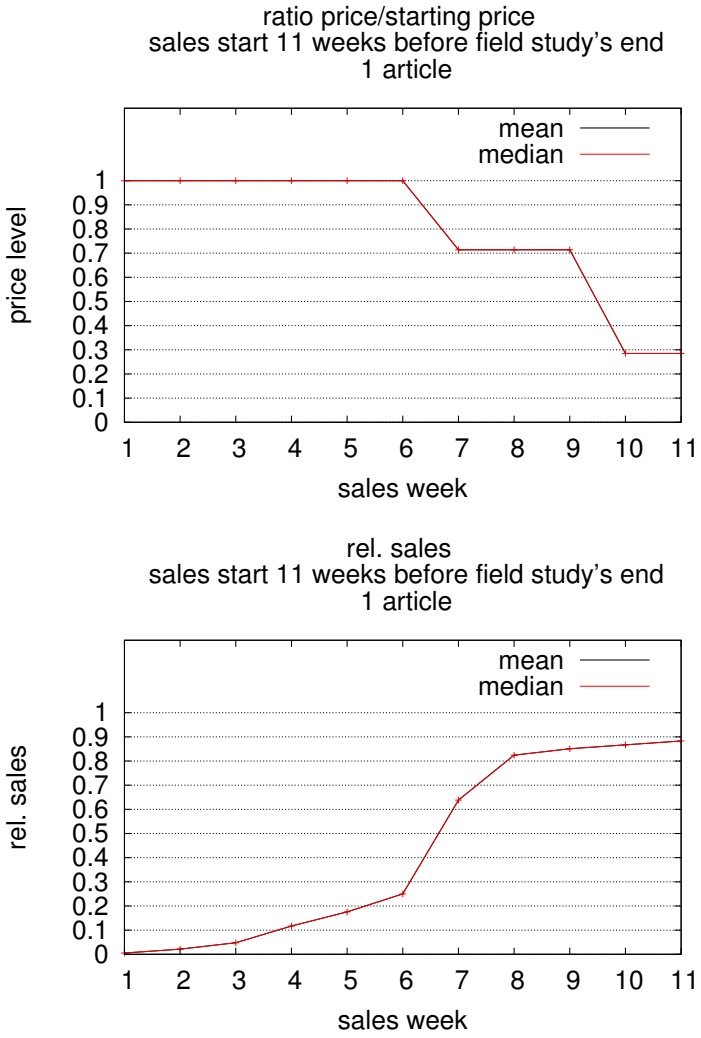

Figure B.2: Effect of mark-downs – 11 weeks selling time

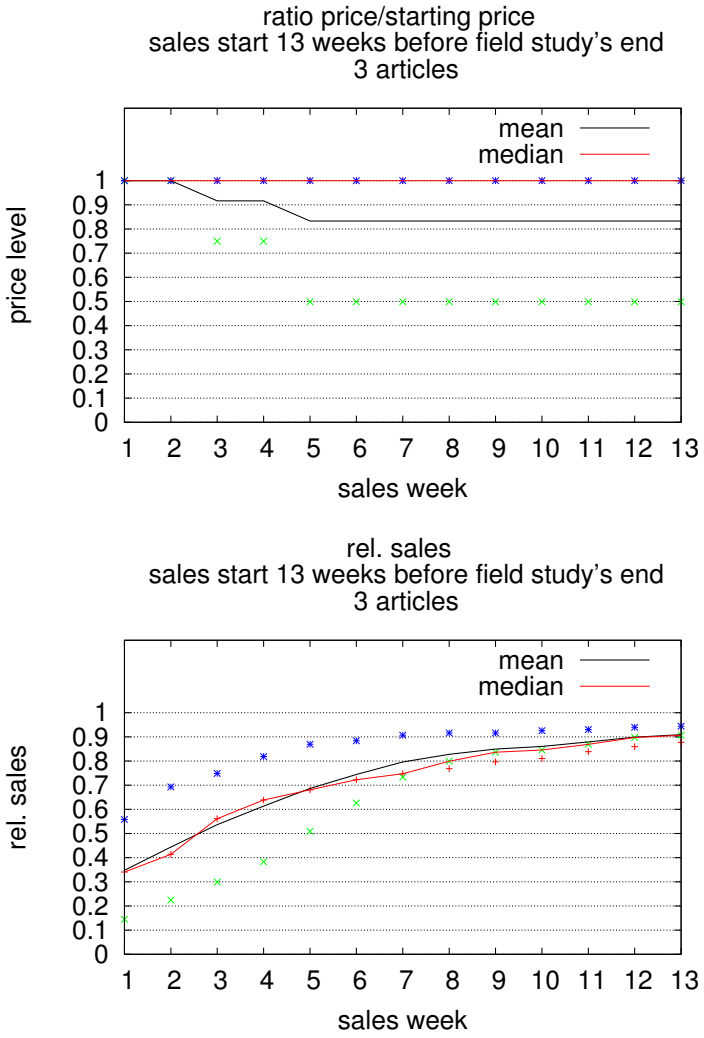

Figure B.3: Effect of mark-downs – 13 weeks selling time

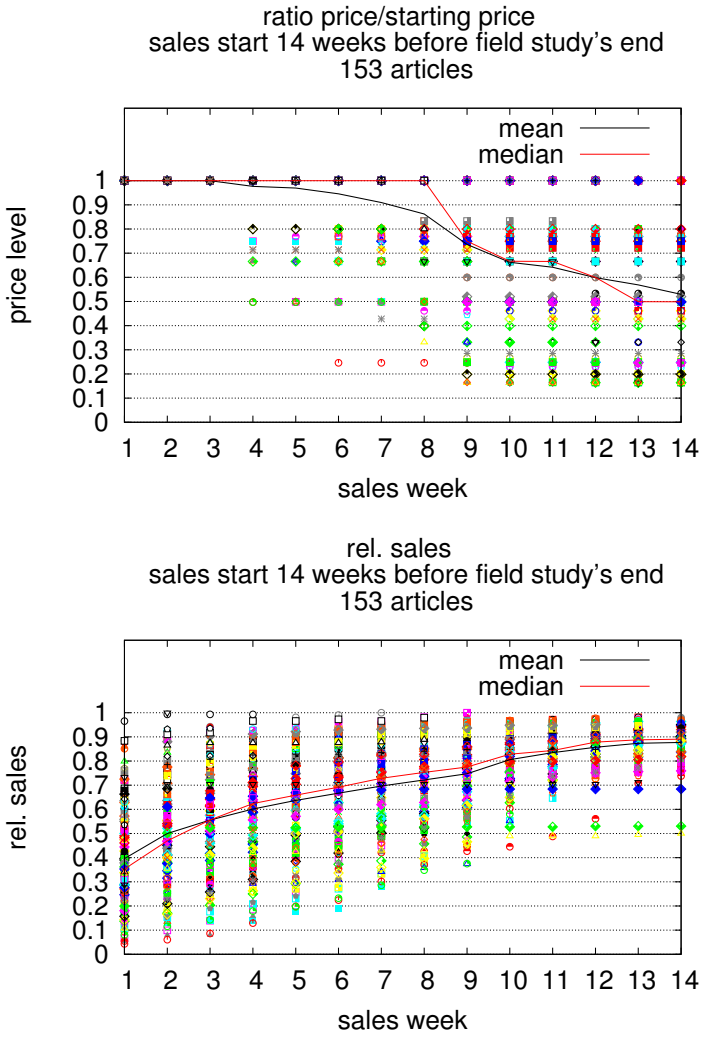

Figure B.4: Effect of mark-downs – 14 weeks selling time

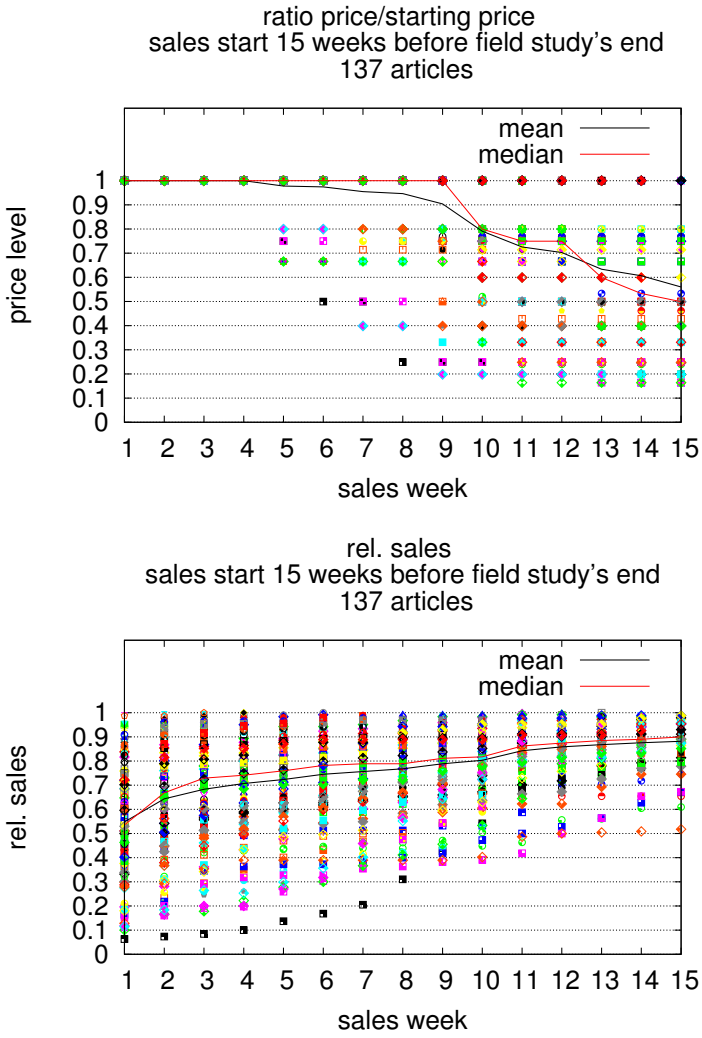

Figure B.5: Effect of mark-downs – 15 weeks selling time

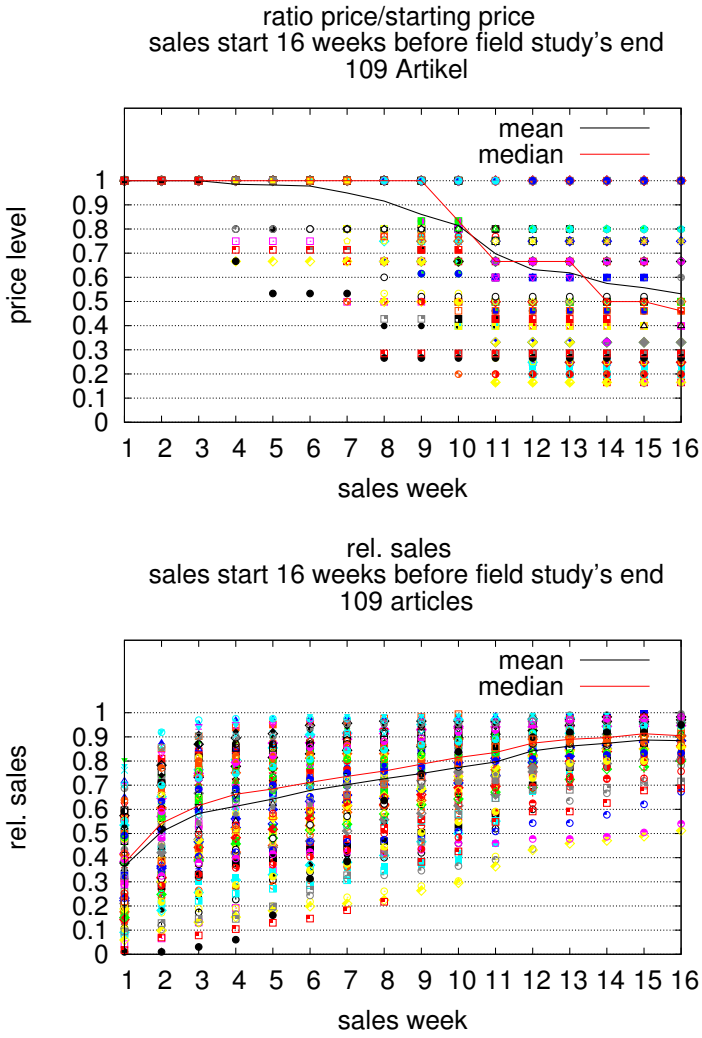

Figure B.6: Effect of mark-downs – 16 weeks selling time

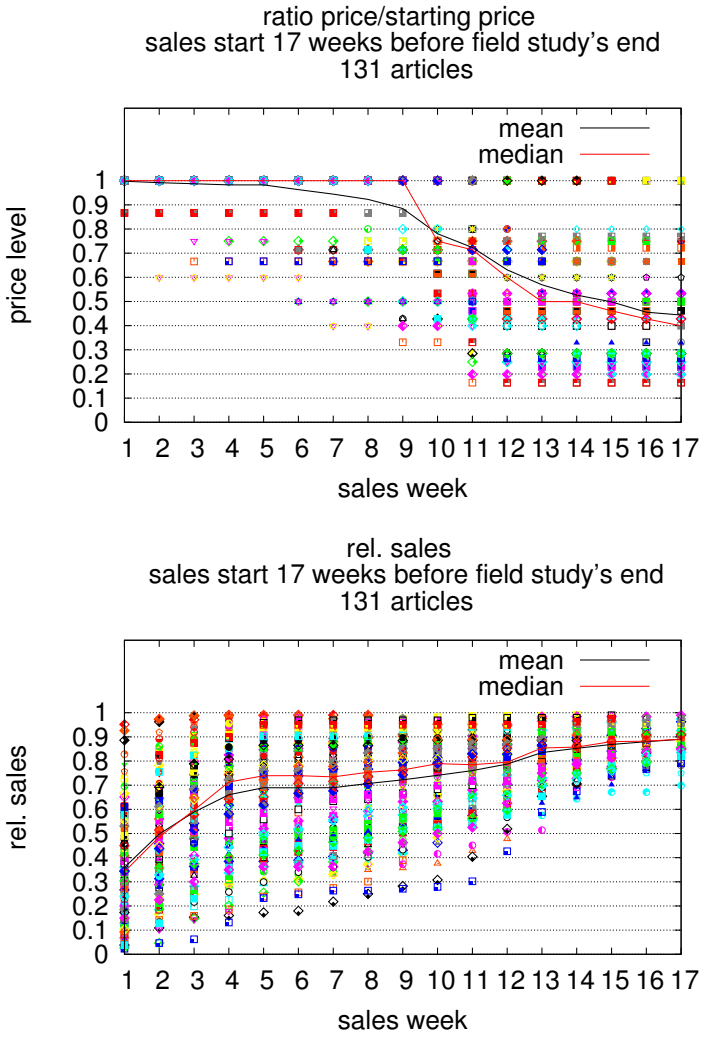

Figure B.7: Effect of mark-downs – 17 weeks selling time

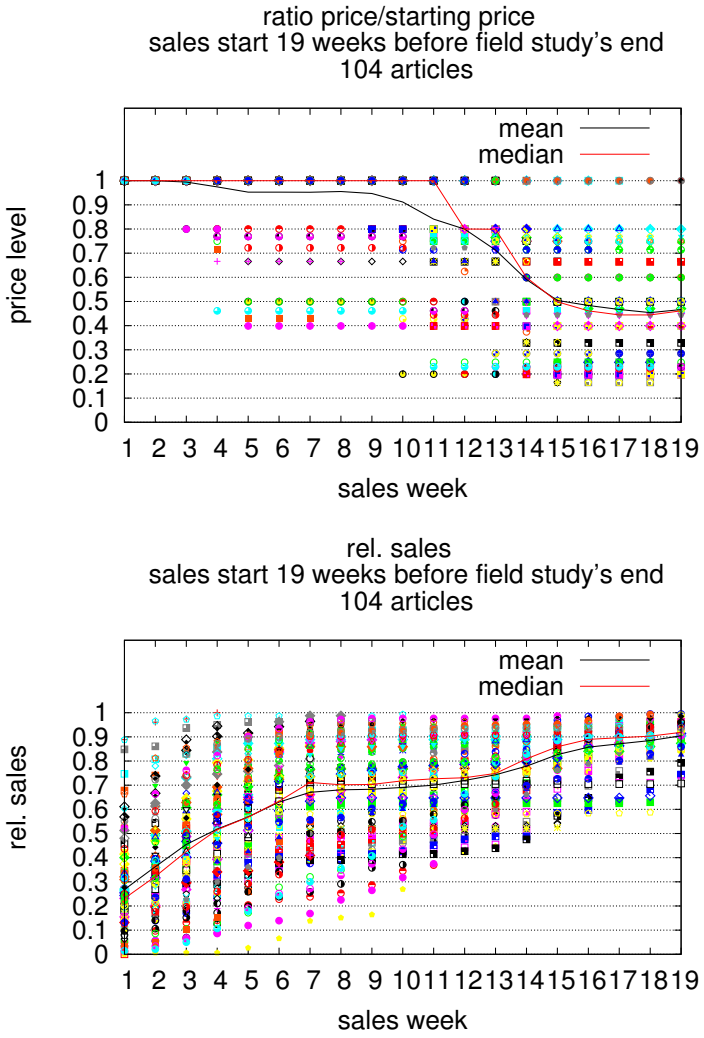

Figure B.8: Effect of mark-downs – 19 weeks selling time

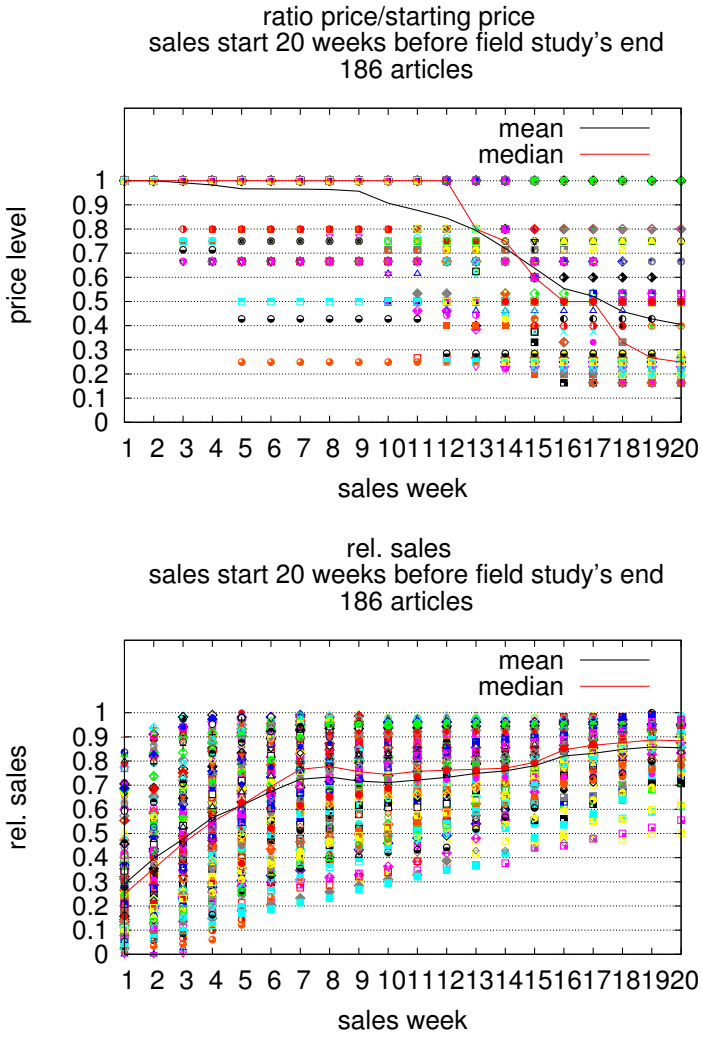

Figure B.9: Effect of mark-downs – 20 weeks selling time

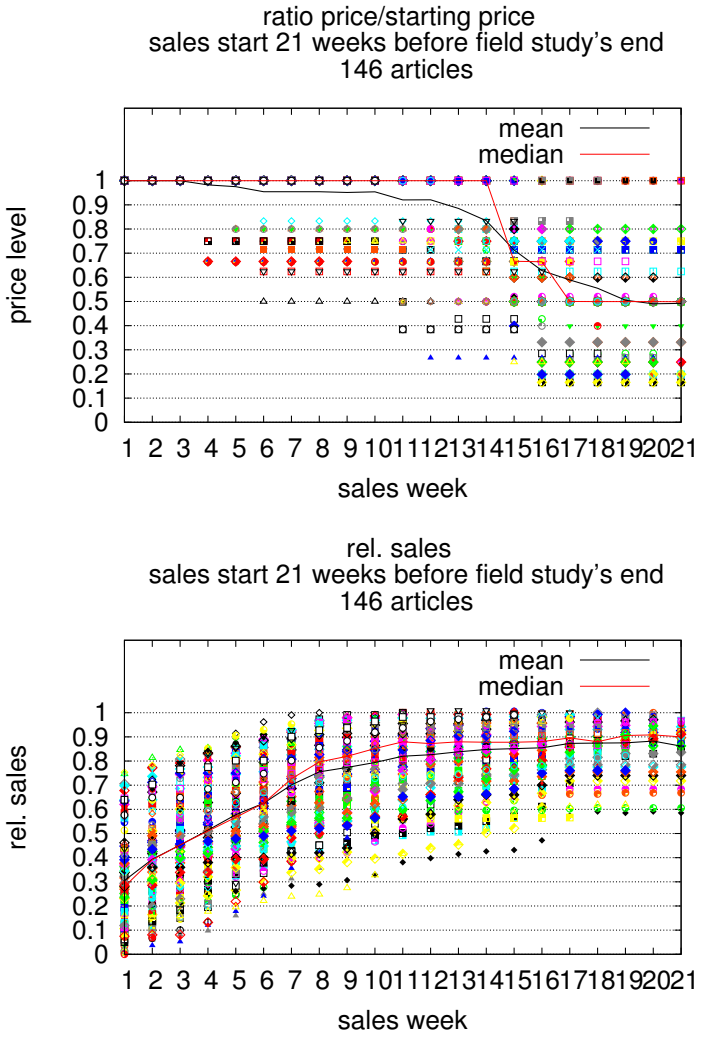

Figure B.10: Effect of mark-downs – 21 weeks selling time

Appendix C

Single supply revenues for the accompanying example

In the following tables we state the single supply revenues $\bar{a}^{e\to t}_{b,s,n}$ for the accompanying example, Section 4.7. The indices for the price trajectories t are given in the row and the number of items in the columns. For demand exceeding supplies, see Observation 2, of more than n items the additional revenue per item is given at the column with label "$n+$".

low scenario, branch 1, size S					
t/n	1	2	3	4	4+
0	10.49	20.84	26.08	26.34	0.26
1	10.49	20.84	26.06	26.22	0.16
2	10.49	20.84	24.41	24.57	0.16
3	10.49	19.59	24.84	25.00	0.16
4	10.49	19.59	22.84	23.20	0.06
5	10.49	18.61	19.96	20.71	0.16

low scenario, branch 1, size L						
t/n	1	2	3	4	5	5+
0	10.49	20.93	31.25	34.98	35.24	0.26
1	10.49	20.93	31.25	33.43	33.59	0.16
2	10.49	20.93	31.25	31.94	32.10	0.16
3	10.49	20.93	28.75	33.49	33.65	0.16
4	10.49	20.93	28.75	31.94	32.00	0.06
5	10.49	20.93	26.79	28.14	28.83	0.16

low scenario, branch 2, size S			
t/n	1	2	2+
0	10.43	17.66	0.26
1	10.43	16.15	0.16
2	10.43	14.66	0.16
3	10.18	14.41	0.16
4	10.18	12.85	0.06
5	9.98	11.21	0.16

low scenario, branch 2, size L				
t/n	1	2	3	3+
0	10.49	20.81	23.05	0.26
1	10.49	20.56	23.00	0.16
2	10.49	20.37	21.29	0.16
3	10.49	18.32	21.51	0.16
4	10.49	18.32	19.72	0.06
5	10.49	16.36	17.63	0.16

normal scenario, branch 1, size S

t/n	1	2	3	4	5	6	7	8	8+
0	10.49	20.98	31.36	41.69	51.91	52.16	52.42	52.68	0.26
1	10.49	20.98	31.36	41.69	49.43	52.12	52.28	52.44	0.16
2	10.49	20.98	31.36	41.69	47.49	48.82	48.98	49.14	0.16
3	10.49	20.98	31.36	39.19	44.46	49.67	49.83	49.99	0.16
4	10.49	20.98	31.36	39.19	44.46	45.69	46.34	46.40	0.06
5	10.49	20.98	31.36	37.23	38.58	39.92	41.25	41.41	0.16

normal scenario, branch 1, size L

t/n	1	2	3	4	5	6	7	8	9	9+
0	10.49	20.98	31.47	41.85	52.23	62.50	69.70	69.96	70.22	0.26
1	10.49	20.98	31.47	41.85	52.23	62.50	66.71	66.87	67.03	0.16
2	10.49	20.98	31.47	41.85	52.23	62.50	63.72	63.88	64.04	0.16
3	10.49	20.98	31.47	41.85	52.23	57.50	62.77	66.98	67.14	0.16
4	10.49	20.98	31.47	41.85	52.23	57.50	62.77	63.89	63.95	0.06
5	10.49	20.98	31.47	41.85	52.23	53.58	54.93	56.28	57.50	0.16

normal scenario, branch 2, size S

t/n	1	2	3	4	4+
0	10.49	20.86	31.10	35.32	0.26
1	10.49	20.86	29.61	32.30	0.16
2	10.49	20.86	28.45	29.31	0.16
3	10.49	20.36	25.62	28.81	0.16
4	10.49	20.36	24.82	25.70	0.06
5	10.49	19.97	21.32	22.42	0.16

normal scenario, branch 2, size L

t/n	1	2	3	4	5	6	6+
0	10.49	20.98	31.36	41.62	45.84	46.10	0.26
1	10.49	20.98	31.36	41.13	45.83	45.99	0.16
2	10.49	20.98	31.36	40.74	42.07	42.58	0.16
3	10.49	20.98	31.36	36.63	41.86	43.03	0.16
4	10.49	20.98	31.36	36.63	38.67	39.44	0.06
5	10.49	20.98	31.36	32.71	34.05	35.27	0.16

high scenario, branch 1, size S

t/n	1	2	3	4	5	6	7	8	9	10	10+
0	10.49	20.98	31.43	41.81	52.14	62.40	67.61	67.87	68.13	68.39	0.26
1	10.49	20.98	31.43	41.81	52.14	61.65	66.87	67.78	67.94	68.10	0.16
2	10.49	20.98	31.43	41.81	52.14	61.07	62.40	63.50	63.66	63.82	0.16
3	10.49	20.98	31.43	41.81	49.89	55.16	60.40	64.61	64.77	64.93	0.16
4	10.49	20.98	31.43	41.81	49.89	55.16	58.41	59.64	60.23	60.29	0.06
5	10.49	20.98	31.43	41.81	48.12	49.47	50.83	52.16	53.49	53.77	0.16

high scenario, branch 1, size L

t/n	1	2	3	4	5	6	7	8	9	10	11	12	12+
0	10.49	20.98	31.47	41.95	52.33	62.71	73.04	83.29	90.58	90.84	91.10	91.36	0.26
1	10.49	20.98	31.47	41.95	52.33	62.71	73.04	82.30	86.70	86.86	87.02	87.18	0.16
2	10.49	20.98	31.47	41.95	52.33	62.71	73.04	81.52	82.82	82.98	83.14	83.30	0.16
3	10.49	20.98	31.47	41.95	52.33	62.71	70.54	75.81	81.08	86.30	87.17	87.33	0.16
4	10.49	20.98	31.47	41.95	52.33	62.71	70.54	75.81	81.08	82.71	83.09	83.15	0.06
5	10.49	20.98	31.47	41.95	52.33	62.71	68.58	69.93	71.28	72.63	73.97	74.80	0.16

high scenario, branch 2, size S

t/n	1	2	3	4	5	5+
0	10.49	20.90	31.23	41.45	45.87	0.26
1	10.49	20.90	31.23	39.02	41.96	0.16
2	10.49	20.90	31.23	37.12	38.07	0.16
3	10.49	20.90	28.58	33.83	37.42	0.16
4	10.49	20.90	28.58	32.39	33.40	0.06
5	10.49	20.90	26.50	27.84	29.11	0.16

	high scenario, branch 2, size L								
t/n	1	2	3	4	5	6	7	8	8+
0	10.49	20.98	31.43	41.80	52.07	59.47	59.73	59.99	0.26
1	10.49	20.98	31.43	41.80	52.07	57.64	59.66	59.82	0.16
2	10.49	20.98	31.43	41.80	52.07	54.03	55.23	55.39	0.16
3	10.49	20.98	31.43	41.30	46.57	51.81	55.81	55.97	0.16
4	10.49	20.98	31.43	41.30	46.57	49.66	50.89	51.28	0.06
5	10.49	20.98	31.43	40.90	42.25	43.60	44.94	45.88	0.16

Appendix D

Demand estimation via logistic regression

	Coef	S.E.	Wald Z	P
$y \geq 1$	-2.99121	0.256641	-11.66	0.0000
$y \geq 2$	-5.72449	0.262209	-21.83	0.0000
$y \geq 3$	-8.23650	0.320577	-25.69	0.0000
$y \geq 4$	-9.23334	0.407111	-22.68	0.0000
at	1.26649	0.063337	20.00	0.0000
we	-0.05551	0.004702	-11.81	0.0000
st	0.83864	0.016701	50.22	0.0000
sp	-0.10739	0.007625	-14.08	0.0000
$\frac{pr}{sp}$	-1.02588	0.092116	-11.14	0.0000
branch=683	0.40555	0.251831	1.61	0.1073
branch=701	0.42276	0.294544	1.44	0.1512
branch=894	-0.06808	0.327038	-0.21	0.8351
branch=1096	0.91454	0.250584	3.65	0.0003
branch=1160	1.14606	0.253172	4.53	0.0000
branch=1224	0.51938	0.297256	1.75	0.0806
branch=1375	0.04119	0.326919	0.13	0.8997
branch=1384	0.62334	0.270225	2.31	0.0211
branch=1395	0.11656	0.331805	0.35	0.7254
branch=1432	0.10299	0.300083	0.34	0.7315
branch=1456	0.42631	0.294990	1.45	0.1484
branch=1484	0.78789	0.251974	3.13	0.0018
branch=1486	0.41850	0.302300	1.38	0.1662
branch=1490	0.40608	0.263508	1.54	0.1233
branch=1527	0.73817	0.260580	2.83	0.0046
branch=1569	0.85198	0.254985	3.34	0.0008
branch=1599	0.55777	0.274892	2.03	0.0425
branch=1687	0.34086	0.295072	1.16	0.2480
branch=1720	0.06354	0.322239	0.20	0.8437
branch=1863	0.68430	0.254801	2.69	0.0072
branch=1882	0.66197	0.265527	2.49	0.0127
branch=1929	0.30273	0.258620	1.17	0.2418
branch=1964	1.00468	0.261322	3.84	0.0001
branch=1991	0.74825	0.256327	2.92	0.0035
branch=2056	0.17367	0.259544	0.67	0.5034
branch=2066	0.65254	0.278156	2.35	0.0190
branch=2093	0.42737	0.267164	1.60	0.1097
branch=2096	0.68954	0.275137	2.51	0.0122
branch=2182	0.24661	0.304765	0.81	0.4184
size=40	0.50225	0.292677	1.72	0.0862
size=44	0.56607	0.277932	2.04	0.0417
branch=683 * size=40	-0.16792	0.332657	-0.50	0.6137
branch=701 * size=40	-0.25824	0.390047	-0.66	0.5079
branch=894 * size=40	-0.42118	0.444107	-0.95	0.3429
branch=1096 * size=40	-0.61760	0.335986	-1.84	0.0660
branch=1160 * size=40	-0.62938	0.338675	-1.86	0.0631
branch=1224 * size=40	-0.80276	0.422501	-1.90	0.0574

branch=1375 * size=40	−0.69846	0.464456	−1.50	0.1326
branch=1384 * size=40	−0.56853	0.366081	−1.55	0.1204
branch=1395 * size=40	−0.27257	0.441727	−0.62	0.5372
branch=1432 * size=40	−0.14160	0.401816	−0.35	0.7245
branch=1456 * size=40	−0.37085	0.396080	−0.94	0.3491
branch=1484 * size=40	−0.54529	0.335403	−1.63	0.1040
branch=1486 * size=40	−0.89554	0.433163	−2.07	0.0387
branch=1490 * size=40	−0.49841	0.351879	−1.42	0.1567
branch=1527 * size=40	−0.23725	0.343948	−0.69	0.4903
branch=1569 * size=40	−0.63781	0.340548	−1.87	0.0611
branch=1599 * size=40	−0.44313	0.367939	−1.20	0.2285
branch=1687 * size=40	−0.82185	0.415250	−1.98	0.0478
branch=1720 * size=40	−0.65954	0.456013	−1.45	0.1481
branch=1863 * size=40	−0.64082	0.341226	−1.88	0.0604
branch=1882 * size=40	−0.58157	0.362610	−1.60	0.1087
branch=1929 * size=40	−0.38308	0.343194	−1.12	0.2643
branch=1964 * size=40	−0.80925	0.353476	−2.29	0.0221
branch=1991 * size=40	−0.70503	0.346181	−2.04	0.0417
branch=2056 * size=40	−0.20091	0.342676	−0.59	0.5577
branch=2066 * size=40	−0.75010	0.379073	−1.98	0.0478
branch=2093 * size=40	−0.28578	0.354499	−0.81	0.4202
branch=2096 * size=40	−0.48101	0.370227	−1.30	0.1939
branch=2182 * size=40	−0.72650	0.424956	−1.71	0.0873
branch=683 * size=44	−0.35055	0.320565	−1.09	0.2742
branch=701 * size=44	−0.74507	0.404248	−1.84	0.0653
branch=894 * size=44	−0.17317	0.424871	−0.41	0.6836
branch=1096 * size=44	−0.31359	0.322606	−0.97	0.3310
branch=1160 * size=44	−0.75593	0.327884	−2.31	0.0211
branch=1224 * size=44	−0.87093	0.412327	−2.11	0.0347
branch=1375 * size=44	−0.56869	0.441446	−1.29	0.1977
branch=1384 * size=44	−0.38354	0.351260	−1.09	0.2749
branch=1395 * size=44	−0.81523	0.458873	−1.78	0.0756
branch=1432 * size=44	−0.10744	0.388336	−0.28	0.7820
branch=1456 * size=44	−0.16553	0.378482	−0.44	0.6619
branch=1484 * size=44	−0.67481	0.328498	−2.05	0.0400
branch=1486 * size=44	−0.69392	0.409814	−1.69	0.0904
branch=1490 * size=44	−0.24879	0.337260	−0.74	0.4607
branch=1527 * size=44	−0.42538	0.334314	−1.27	0.2032
branch=1569 * size=44	−0.50459	0.327853	−1.54	0.1238
branch=1599 * size=44	−0.19827	0.348966	−0.57	0.5699
branch=1687 * size=44	−0.80178	0.385743	−2.08	0.0377
branch=1720 * size=44	−0.19315	0.417738	−0.46	0.6438
branch=1863 * size=44	−0.74270	0.330676	−2.25	0.0247
branch=1882 * size=44	−0.59922	0.352444	−1.70	0.0891
branch=1929 * size=44	−0.27425	0.332048	−0.83	0.4088
branch=1964 * size=44	−0.66239	0.338624	−1.96	0.0504
branch=1991 * size=44	−0.42093	0.331364	−1.27	0.2040
branch=2056 * size=44	−0.24034	0.331231	−0.73	0.4681
branch=2066 * size=44	−0.62307	0.366819	−1.70	0.0894
branch=2093 * size=44	−0.83065	0.344630	−2.41	0.0159
branch=2096 * size=44	−0.45605	0.356120	−1.28	0.2003
branch=2182 * size=44	−0.74137	0.412177	−1.80	0.0721

Appendix E

Instances

In terms of computational tests we created real instances from several CSV-files we obtained from our industrial partner. There were different CSV-files containing the related commodity group, supplied branches and sizes, benchmark data as starting price and remaining price steps, mark-down costs, overall supply, bounds for the lot-types, etc.. We used historical transaction data for demand estimation, Chapter 3. We standardized input by condensing all data concerning an an article/instance in an XML-file. As well the exact solver ISPO-BAB as our heuristic ISPO-PingPong use the XML-file as only input. For price optimization on its own, POP-DYN, additionally the supply per branch and size in the form of an CSV-file has to be provided.

We denote the set of real instances by $\mathcal{I}$. The benchmark data for all of these instances is outlined in the tables E.1 to E.17. We always consider the three scenarios "low seller", "normal seller" and "high seller'. Their probabilities are given in the last three columns of the tables. The observation time for all these instances amounts to $k_{\text{obs}} = 2$, the number of sales periods is given by $|K| = 14$, i.e. $k_{\max} = 13$ (3 months or 13 weeks is the regular selling time per product at our industrial partner). The set of multiplicities is given by $M = \{1, 2, 3\}$.

We consider lot-types with one item at minimum $v^{\min} = 1$ and three items at maximum per size and per branch, $v^{\max} = 3$. The minimum of items per lot-type amounts to $vl^{\min} = 6$, the maximum $vl^{\max} = 18$. For example for the instances with $|S| = 6$ this yields 729 different lot-types.

From the instances with 6 sizes $|S| = 6$ for some tests we created smaller test instances by only considering the first 30 branches in the order as they appear in the corresponding XML-file. For these smaller instances we consider only 5 periods: $|K| = 5$ or $k_{\max} = 4$. The remaining data is as given. We set the bounds for overall supply to $\underline{I} = 210$ and $\bar{I} = 240$. All lot-types with at least one item per size and at most three items per size are considered. The number of items in a lot-type lies between 6 and 12. This yields 435 different lot-types. This set of instances is denoted by $\mathcal{I}_6^{\text{test}}$.

no.	κ	$\underline{I}$	$\bar{I}$	$\|S\|$	$\|B\|$	Prob(low)	Prob(normal)	Prob(high)
1	4	14256	14544	6	1626	0,37	0,28	0,35
2	4	14256	14544	6	1626	0,44	0,28	0,28
3	4	14256	14544	6	1626	0,37	0,28	0,35
4	4	14256	14544	6	1626	0,37	0,28	0,35
5	4	14256	14544	6	1626	0,25	0,36	0,39
6	4	14256	14544	6	1626	0,25	0,36	0,39
7	4	14256	14544	7	1626	0,16	0,38	0,46
8	4	11088	11312	5	1626	0,54	0,22	0,24
9	4	11088	11312	5	1626	0,54	0,22	0,24
10	4	11088	11312	5	1626	0,78	0,22	0
11	4	11088	11312	5	1626	0,54	0,22	0,24
12	4	11088	11312	5	1626	0,54	0,22	0,24
13	4	11088	11312	5	1626	0,54	0,22	0,24
14	4	9405	9595	3	1626	0,07	0,3	0,64
15	4	9405	9595	3	1626	0,16	0,4	0,44
16	4	7524	7676	4	1626	0	0,08	0,92
17	4	7524	7676	4	1626	0	0,08	0,92
18	4	9306	9494	3	1626	0,22	0,43	0,35
19	4	9306	9494	3	1626	0,22	0,43	0,35
20	4	13167	13433	3	1626	0,22	0,43	0,35
21	4	11286	11514	3	1626	0,22	0,43	0,35
22	4	11286	11514	3	1626	0,22	0,43	0,35
23	4	11286	11514	3	1626	0,22	0,43	0,35
24	4	11286	11514	3	1626	0,22	0,43	0,35
25	4	11088	11312	3	1626	0,22	0,43	0,35
26	4	11088	11312	3	1626	0,22	0,43	0,35
27	4	9504	9696	3	1626	0,22	0,43	0,35
28	2	6336	6464	3	1626	0,32	0,38	0,3
29	2	6336	6464	3	1626	0,32	0,38	0,3
30	4	7920	8080	3	1626	0,33	0,3	0,37
31	4	9356	9544	4	1370	0,33	0,3	0,37
32	4	12474	12726	4	1626	0,33	0,3	0,37
33	4	11088	11312	5	1626	0,33	0,3	0,37
34	4	11088	11312	5	1626	0,33	0,3	0,37
35	4	9504	9696	4	1626	0,21	0,37	0,42
36	4	9504	9696	4	1626	0,21	0,37	0,42
37	4	9504	9696	4	1626	0,34	0,38	0,29
38	4	9504	9696	4	1626	0,34	0,38	0,29
39	2	7920	8080	4	1626	0,34	0,38	0,29
40	2	7920	8080	4	1626	0,34	0,38	0,29
41	2	7920	8080	4	1626	0,34	0,38	0,29
42	3	9504	9696	4	1626	0,32	0,29	0,38
43	3	9504	9696	4	1626	0,32	0,29	0,38
44	4	14256	14544	6	1626	0,37	0,28	0,35
45	4	16929	17271	6	1626	0,18	0,31	0,51
46	4	16929	17271	6	1626	0,18	0,31	0,51
47	4	16929	17271	6	1626	0,18	0,31	0,51
48	4	16929	17271	6	1626	0,16	0,38	0,46
49	4	16929	17271	6	1626	0,34	0,24	0,42
50	4	16929	17271	6	1626	0,16	0,38	0,46

Table E.1: Real instances – part 1

no.	κ	$\underline{I}$	$\bar{I}$	$\|S\|$	$\|B\|$	Prob(low)	Prob(normal)	Prob(high)
51	4	16929	17271	6	1626	0,16	0,38	0,46
52	4	16929	17271	6	1626	0,16	0,38	0,46
53	4	16929	17271	6	1626	0,25	0,36	0,39
54	4	15840	16160	7	1626	0,38	0,34	0,28
55	3	15840	16160	7	1626	0,34	0,32	0,34
56	3	15840	16160	7	1626	0,38	0,34	0,28
57	4	15840	16160	7	1626	0,49	0,32	0,19
58	3	15840	16160	7	1626	0,34	0,32	0,34
59	4	11286	11514	4	1626	0,37	0,2	0,43
60	4	11286	11514	4	1626	0,42	0,3	0,28
61	4	16929	17271	6	1626	0,34	0,33	0,34
62	4	16929	17271	6	1626	0,25	0,26	0,5
63	4	16929	17271	6	1626	0,37	0,28	0,35
64	4	16929	17271	6	1626	0,37	0,28	0,35
65	4	16929	17271	6	1626	0,37	0,28	0,35
66	4	16929	17271	6	1626	0,37	0,28	0,35
67	4	16929	17271	6	1626	0,25	0,26	0,5
68	4	13167	13433	5	1626	0,33	0,33	0,34
69	4	13167	13433	5	1626	0,33	0,33	0,34
70	4	13167	13433	5	1626	0,33	0,33	0,34
71	4	13167	13433	5	1626	0,33	0,33	0,34
72	4	13167	13433	5	1626	0,33	0,33	0,34
73	4	13167	13433	5	1626	0,33	0,33	0,34
74	2	11088	11312	5	1626	0,33	0,33	0,34
75	2	11088	11312	5	1626	0,33	0,33	0,34
76	2	11088	11312	5	1626	0,45	0,29	0,26
77	2	11088	11312	5	1626	0,45	0,29	0,26
78	2	11088	11312	5	1626	0,45	0,29	0,26
79	2	11088	11312	5	1626	0,45	0,29	0,26
80	4	13167	13433	5	1626	0,54	0,22	0,24
81	4	13167	13433	5	1626	0,54	0,22	0,24
82	4	13167	13433	5	1626	0,54	0,22	0,24
83	4	15840	16160	7	1626	0,49	0,32	0,19
84	4	19305	19695	7	1626	0,38	0,34	0,28
85	4	19305	19695	7	1626	0,38	0,34	0,28
86	4	19305	19695	7	1626	0,38	0,34	0,28
87	4	19305	19695	7	1626	0,38	0,34	0,28
88	4	19305	19695	7	1626	0,38	0,34	0,28
89	4	19305	19695	7	1626	0,38	0,34	0,28
90	4	18810	19190	7	1626	0,38	0,34	0,28
91	4	19305	19695	7	1626	0,38	0,34	0,28
92	4	9306	9494	3	1626	0,22	0,43	0,35
93	4	11286	11514	3	1626	0,22	0,43	0,35
94	4	9504	9696	3	1626	0,22	0,43	0,35
95	4	9306	9494	3	1626	0,22	0,43	0,35
96	3	7920	8080	2	1626	0,22	0,43	0,35
97	4	9306	9494	3	1626	0,32	0,38	0,3
98	4	16929	17271	6	1626	0,32	0,38	0,3
99	2	6336	6464	3	1626	0,32	0,38	0,3
100	2	6336	6464	3	1626	0,32	0,38	0,3

Table E.2: Real instances – part 2

no.	κ	$\underline{I}$	$\bar{I}$	$\lvert S\rvert$	$\lvert B\rvert$	Prob(low)	Prob(normal)	Prob(high)
101	4	10395	10605	3	1626	0,21	0,5	0,29
102	4	10395	10605	3	1626	0,21	0,5	0,29
103	4	10395	10605	3	1626	0,21	0,5	0,29
104	4	10395	10605	4	1626	0,21	0,5	0,29
105	3	6336	6464	4	1636	0,21	0,5	0,29
106	4	12474	12726	4	1626	0,33	0,3	0,37
107	4	12474	12726	4	1626	0,33	0,3	0,37
108	4	10395	10605	3	1626	0,33	0,3	0,37
109	4	10395	10605	3	1626	0,33	0,3	0,37
110	4	10395	10605	3	1626	0,5	0,31	0,2
111	4	10395	10605	3	1626	0,5	0,31	0,2
112	4	9504	9696	4	1626	0,34	0,38	0,29
113	4	9504	9696	4	1626	0,34	0,38	0,29
114	4	9504	9696	4	1626	0,34	0,38	0,29
115	4	9504	9696	4	1626	0,34	0,38	0,29
116	4	12474	12726	4	1626	0,21	0,37	0,42
117	4	12474	12726	4	1626	0,21	0,37	0,42
118	4	12474	12726	4	1626	0,32	0,29	0,38
119	4	11286	11514	4	1626	0,32	0,5	0,18
120	4	11286	11514	4	1626	0,33	0,25	0,42
121	4	11286	11514	4	1626	0,38	0,33	0,29
122	4	11286	11514	4	1626	0,26	0,26	0,48
123	4	9504	9696	4	1626	0,33	0,34	0,33
124	4	14256	14544	6	1626	0,37	0,28	0,35
125	4	16929	17271	6	1626	0,25	0,36	0,39
126	4	16929	17271	6	1626	0,16	0,38	0,46
127	4	16929	17271	6	1626	0,16	0,38	0,46
128	4	16929	17271	6	1626	0,25	0,36	0,39
129	4	16929	17271	6	1626	0,16	0,38	0,46
130	4	16929	17271	6	1626	0,18	0,31	0,51
131	4	16929	17271	6	1626	0,18	0,31	0,51
132	4	11088	11312	5	1626	0,6	0,08	0,32
133	4	15840	16160	7	1626	0,39	0,3	0,31
134	4	11286	11514	3	1626	0	0,3	0,7
135	4	9504	9696	3	1626	0,22	0,43	0,35
136	4	11088	11312	3	1626	0,22	0,43	0,35
137	4	14256	14544	6	1626	0,32	0,38	0,3
138	4	10395	10605	4	1626	0	0	1
139	4	13167	13433	5	1636	0,44	0,28	0,28
140	4	16929	17271	6	1636	0,44	0,28	0,28
141	4	16929	17271	6	1636	0,34	0,33	0,34
142	4	16929	17271	6	1636	0,37	0,28	0,35
143	4	16929	17271	6	1636	0,2	0,48	0,32
144	4	16929	17271	6	1636	0,25	0,26	0,5
145	4	16929	17271	6	1636	0,25	0,26	0,5
146	4	13167	13433	5	1636	0,33	0,33	0,34
147	4	13167	13433	5	1636	0,33	0,33	0,34
148	2	13167	13433	5	1636	0,45	0,29	0,26
149	4	13167	13433	5	1636	0,36	0,33	0,31
150	4	13167	13433	5	1636	0,54	0,22	0,24

Table E.3: Real instances – part 3

no.	κ	$\underline{I}$	$\bar{I}$	$\lvert S\rvert$	$\lvert B\rvert$	Prob(low)	Prob(normal)	Prob(high)
151	4	13167	13433	5	1636	0,78	0,22	0
152	4	13167	13433	5	1636	0,47	0,26	0,27
153	4	13167	13433	5	1636	0,54	0,22	0,24
154	4	13167	13433	5	1636	0,56	0,18	0,26
155	4	13167	13433	5	1636	0,49	0,32	0,19
156	4	15840	16160	7	1636	0,34	0,32	0,34
157	4	13167	13433	5	1636	0,56	0,18	0,26
158	4	15840	16160	7	1636	0,38	0,34	0,28
159	4	13167	13433	5	1636	0,38	0,34	0,28
160	4	13167	13433	5	1636	0,34	0,32	0,34
161	4	9405	9595	3	1636	0,07	0,3	0,64
162	4	9405	9595	3	1636	0,16	0,4	0,44
163	4	7524	7676	4	1636	0	0,08	0,92
164	4	7524	7676	4	1636	0	0,08	0,92
165	4	9405	9595	3	1636	0,22	0,43	0,35
166	4	9405	9595	3	1636	0,22	0,43	0,35
167	4	11088	11312	3	1636	0,22	0,43	0,35
168	4	11088	11312	3	1636	0,22	0,43	0,35
169	2	6336	6464	3	1636	0,22	0,43	0,35
170	4	5643	5757	3	1636	0,32	0,38	0,3
171	4	12474	12726	4	1636	0,33	0,3	0,37
172	4	12474	12726	4	1636	0,33	0,3	0,37
173	4	12474	12726	4	1636	0,33	0,3	0,37
174	4	9504	9696	4	1636	0,33	0,3	0,37
175	4	9504	9696	4	1636	0,33	0,3	0,37
176	4	9504	9696	4	1636	0,33	0,3	0,37
177	4	9504	9696	4	1636	0,33	0,3	0,37
178	4	9504	9696	4	1636	0,21	0,37	0,42
179	4	9504	9696	4	1636	0,21	0,37	0,42
180	4	9504	9696	4	1636	0,21	0,37	0,42
181	4	9504	9696	4	1636	0,21	0,37	0,42
182	4	13167	13433	5	1636	0,34	0,38	0,29
183	4	13167	13433	5	1636	0,34	0,38	0,29
184	4	10395	10605	5	1636	0,32	0,29	0,38
185	4	10395	10605	5	1636	0,32	0,29	0,38
186	4	11286	11514	4	1636	0,38	0,33	0,29
187	4	11286	11514	4	1636	0,33	0,25	0,42
188	4	11286	11514	4	1636	0,33	0,25	0,42
189	4	11286	11514	4	1636	0,38	0,33	0,29
190	4	11286	11514	4	1636	0,32	0,5	0,18
191	4	9504	9696	4	1636	0,38	0,33	0,29
192	4	16929	17271	6	1636	0,34	0,24	0,42
193	4	13167	13433	6	1636	0,47	0,29	0,24
194	4	16929	17271	6	1636	0,18	0,31	0,51
195	4	16929	17271	6	1636	0,18	0,31	0,51
196	4	16929	17271	6	1636	0,16	0,38	0,46
197	4	16929	17271	6	1636	0,16	0,38	0,46
198	4	16929	17271	6	1636	0,16	0,38	0,46
199	4	16929	17271	6	1636	0,16	0,38	0,46
200	4	16929	17271	6	1636	0,25	0,36	0,39

Table E.4: Real instances – part 4

no.	κ	$\underline{I}$	$\bar{I}$	$\lvert S\rvert$	$\lvert B\rvert$	Prob(low)	Prob(normal)	Prob(high)
201	4	13167	13433	5	1636	0,33	0,33	0,34
202	4	13167	13433	5	1636	0,33	0,33	0,34
203	4	13167	13433	5	1636	0,22	0,36	0,41
204	4	13167	13433	5	1636	0,22	0,36	0,41
205	4	13167	13433	5	1636	0,33	0,33	0,34
206	4	13167	13433	5	1636	0,33	0,33	0,34
207	4	13167	13433	5	1636	0,47	0,26	0,27
208	4	13167	13433	5	1636	0,54	0,22	0,24
209	4	9504	9696	3	1636	0,22	0,43	0,35
210	4	11088	11312	3	1636	0,22	0,43	0,35
211	4	11088	11312	3	1636	0,22	0,43	0,35
212	4	11088	11312	3	1636	0,22	0,43	0,35
213	3	9504	9696	4	1636	0,21	0,5	0,29
214	3	7920	8080	4	1636	0,21	0,5	0,29
215	3	7920	8080	4	1636	0,21	0,5	0,29
216	4	9504	9696	4	1636	0,33	0,3	0,37
217	4	12672	12928	4	1636	0,33	0,3	0,37
218	4	6336	6464	3	1636	0,5	0,31	0,2
219	4	6336	6464	3	1636	0,5	0,31	0,2
220	4	6336	6464	3	1636	0,5	0,31	0,2
221	4	6336	6464	3	1636	0,5	0,31	0,2
222	4	11286	11514	4	1636	0,21	0,37	0,42
223	4	11286	11514	4	1636	0,21	0,37	0,42
224	4	11286	11514	4	1636	0,21	0,37	0,42
225	4	11286	11514	4	1636	0,21	0,37	0,42
226	4	11286	11514	4	1636	0,34	0,38	0,29
227	4	11286	11514	4	1636	0,34	0,38	0,29
228	4	11286	11514	4	1636	0,34	0,38	0,29
229	4	10395	10605	4	1636	0,32	0,29	0,38
230	4	10395	10605	4	1636	0,32	0,29	0,38
231	4	13167	13433	5	1636	0,35	0,39	0,26
232	4	13167	13433	5	1636	0,33	0,34	0,33
233	4	13167	13433	5	1636	0,33	0,34	0,33
234	4	13167	13433	5	1636	0,33	0,34	0,33
235	4	13167	13433	5	1636	0,42	0,3	0,28
236	4	14256	14544	6	1636	0,37	0,28	0,35
237	4	14256	14544	6	1636	0,47	0,29	0,24
238	4	19305	19695	7	1636	0,38	0,34	0,28
239	4	19305	19695	7	1636	0,49	0,32	0,19
240	4	19305	19695	7	1636	0,38	0,34	0,28
241	4	18810	19190	7	1636	0,49	0,32	0,19
242	4	19305	19695	7	1636	0,38	0,34	0,28
243	4	15840	16160	7	1636	0,39	0,3	0,31
244	4	19305	19695	7	1636	0,38	0,34	0,28
245	4	19305	19695	7	1636	0,38	0,34	0,28
246	4	18810	19190	7	1636	0,49	0,32	0,19
247	4	9405	9595	3	1636	0,07	0,3	0,64
248	4	9405	9595	3	1636	0,07	0,3	0,64
249	4	9405	9595	3	1636	0,07	0,3	0,64
250	4	9405	9595	3	1636	0,07	0,3	0,64

Table E.5: Real instances – part 5

no.	κ	$\underline{I}$	$\bar{I}$	$\|S\|$	$\|B\|$	Prob(low)	Prob(normal)	Prob(high)
251	4	9405	9595	3	1636	0,07	0,3	0,64
252	4	9405	9595	3	1636	0,07	0,3	0,64
253	4	9405	9595	3	1636	0,07	0,3	0,64
254	4	9405	9595	3	1636	0,16	0,4	0,44
255	4	9405	9595	3	1636	0,22	0,43	0,35
256	4	9405	9595	3	1636	0,22	0,43	0,35
257	4	11088	11312	3	1636	0,22	0,43	0,35
258	4	11088	11312	3	1636	0,22	0,43	0,35
259	2	6336	6464	3	1636	0,22	0,43	0,35
260	4	9405	9595	3	1636	0,32	0,38	0,3
261	4	12474	12726	3	1636	0,33	0,3	0,37
262	4	14553	14847	3	1636	0,33	0,3	0,37
263	4	12474	12726	3	1636	0,33	0,3	0,37
264	4	14553	14847	3	1636	0,33	0,3	0,37
265	4	14256	14544	6	1636	0,37	0,28	0,35
266	4	15840	16160	7	1636	0,25	0,36	0,39
267	4	15840	16160	7	1636	0,16	0,38	0,46
268	3	15840	16160	7	1636	0,38	0,34	0,28
269	4	12672	12928	4	1636	0,22	0,43	0,35
270	4	11088	11312	5	1636	0,42	0,3	0,28
271	4	16929	17271	6	1646	0,34	0,33	0,34
272	4	13167	13433	5	1646	0,44	0,28	0,28
273	4	16929	17271	6	1646	0,25	0,26	0,5
274	4	16929	17271	6	1646	0,37	0,28	0,35
275	4	16929	17271	6	1646	0,2	0,48	0,32
276	4	16929	17271	6	1646	0,37	0,28	0,35
277	4	16929	17271	6	1646	0,2	0,44	0,36
278	4	16929	17271	6	1646	0,1	0,43	0,48
279	4	16929	17271	6	1646	0,37	0,28	0,35
280	4	16929	17271	6	1646	0,34	0,24	0,42
281	4	16929	17271	6	1646	0,18	0,31	0,51
282	4	16929	17271	6	1646	0,34	0,24	0,42
283	4	16929	17271	6	1646	0,16	0,38	0,46
284	4	16929	17271	6	1646	0,16	0,38	0,46
285	4	16929	17271	6	1646	0,25	0,36	0,39
286	4	16929	17271	6	1646	0,16	0,38	0,46
287	4	16929	17271	6	1646	0,16	0,38	0,46
288	4	16929	17271	6	1646	0,16	0,38	0,46
289	4	13167	13433	5	1646	0,47	0,29	0,24
290	4	16929	17271	6	1646	0,25	0,36	0,39
291	4	16929	17271	6	1646	0,16	0,38	0,46
292	4	13167	13433	5	1646	0	0,13	0,87
293	4	16929	17271	6	1646	0,18	0,31	0,51
294	2	11088	11312	5	1646	0,33	0,33	0,34
295	4	11088	11312	5	1646	0,54	0,22	0,24
296	4	10395	10605	7	1646	0,49	0,32	0,19
297	3	15840	16160	7	1646	0,38	0,34	0,28
298	4	11286	11514	3	1646	0,22	0,43	0,35
299	4	16929	17271	6	1646	0,32	0,38	0,3
300	4	7524	7676	3	1646	0,32	0,38	0,3

Table E.6: Real instances – part 6

no.	κ	$\underline{I}$	$\bar{I}$	$\lvert S\rvert$	$\lvert B\rvert$	Prob(low)	Prob(normal)	Prob(high)
301	3	9405	9595	3	1646	0,32	0,38	0,3
302	4	12474	12726	4	1646	0,21	0,5	0,29
303	4	12474	12726	4	1646	0,21	0,5	0,29
304	4	10395	10605	3	1646	0,21	0,5	0,29
305	4	10395	10605	3	1646	0,21	0,5	0,29
306	4	14553	14847	5	1646	0,33	0,3	0,37
307	4	10395	10605	5	1646	0,33	0,3	0,37
308	4	14553	14847	5	1646	0,33	0,3	0,37
309	4	10395	10605	5	1646	0,33	0,3	0,37
310	4	9504	9696	4	1646	0,33	0,3	0,37
311	4	11088	11312	4	1646	0,33	0,3	0,37
312	4	12474	12726	3	1646	0,5	0,31	0,2
313	4	12474	12726	3	1646	0,5	0,31	0,2
314	4	9504	9696	4	1646	0,21	0,37	0,42
315	4	9504	9696	4	1646	0,21	0,37	0,42
316	4	9504	9696	4	1646	0,34	0,38	0,29
317	4	9504	9696	4	1646	0,34	0,38	0,29
318	4	13167	13433	5	1646	0,35	0,39	0,26
319	4	13167	13433	5	1646	0,33	0,34	0,33
320	4	11286	11514	4	1646	0,33	0,34	0,33
321	4	11286	11514	4	1646	0,33	0,34	0,33
322	4	13167	13433	5	1646	0,42	0,3	0,28
323	4	14256	14544	6	1646	0,37	0,28	0,35
324	4	13514	13786	5	1646	0,39	0,3	0,31
325	4	13514	13786	5	1646	0,39	0,3	0,31
326	4	13167	13433	5	1646	0,49	0,32	0,19
327	4	13514	13786	5	1646	0,38	0,34	0,28
328	4	13167	13433	5	1646	0,38	0,34	0,28
329	4	11286	11514	3	1646	0,22	0,43	0,35
330	4	11633	11867	3	1646	0,22	0,43	0,35
331	4	9306	9494	3	1646	0	0,3	0,7
332	4	7920	8080	2	1646	0,22	0,43	0,35
333	4	11088	11312	3	1646	0,22	0,43	0,35
334	4	11088	11312	3	1646	0,22	0,43	0,35
335	3	7524	7676	3	1646	0,32	0,38	0,3
336	3	7524	7676	3	1646	0,32	0,38	0,3
337	4	9405	9595	3	1646	0,32	0,38	0,3
338	2	7920	8080	4	1646	0,21	0,37	0,42
339	2	7920	8080	4	1646	0,21	0,37	0,42
340	2	11088	11312	6	1646	0,34	0,38	0,29
341	2	11088	11312	6	1646	0,34	0,38	0,29
342	4	16929	17271	6	1646	0,25	0,26	0,5
343	4	16929	17271	6	1646	0,2	0,48	0,32
344	4	16929	17271	6	1646	0,34	0,33	0,34
345	4	16929	17271	6	1646	0,2	0,48	0,32
346	4	16929	17271	6	1646	0,25	0,26	0,5
347	4	16929	17271	6	1646	0,37	0,28	0,35
348	4	16929	17271	6	1646	0,37	0,28	0,35
349	4	16929	17271	6	1646	0,37	0,28	0,35
350	4	13167	13433	5	1646	0,47	0,29	0,24

Table E.7: Real instances – part 7

no.	κ	$\underline{I}$	$\bar{I}$	$\lvert S\rvert$	$\lvert B\rvert$	Prob(low)	Prob(normal)	Prob(high)
351	4	16929	17271	6	1646	0,34	0,24	0,42
352	4	16929	17271	6	1646	0,16	0,38	0,46
353	4	16929	17271	6	1646	0,16	0,38	0,46
354	4	16929	17271	6	1646	0,34	0,24	0,42
355	4	13167	13433	5	1646	0,18	0,31	0,51
356	4	13167	13433	5	1646	0,25	0,36	0,39
357	4	13167	13433	5	1646	0,18	0,31	0,51
358	4	13167	13433	5	1646	0,25	0,36	0,39
359	4	16929	17271	6	1646	0,16	0,38	0,46
360	4	13167	13433	5	1646	0,25	0,36	0,39
361	4	13167	13433	5	1646	0,18	0,31	0,51
362	4	16929	17271	6	1646	0,25	0,36	0,39
363	4	19305	19695	7	1646	0,38	0,34	0,28
364	4	19305	19695	7	1646	0,49	0,32	0,19
365	4	19305	19695	7	1646	0,38	0,34	0,28
366	4	19305	19695	7	1646	0,39	0,3	0,31
367	4	19305	19695	7	1646	0,34	0,32	0,34
368	4	19305	19695	7	1646	0,38	0,34	0,28
369	4	19305	19695	7	1646	0,49	0,32	0,19
370	4	9405	9595	3	1646	0,07	0,3	0,64
371	4	9405	9595	3	1646	0,07	0,3	0,64
372	4	9405	9595	3	1646	0,07	0,3	0,64
373	4	9405	9595	3	1646	0,16	0,4	0,44
374	4	9405	9595	3	1646	0,16	0,4	0,44
375	4	9405	9595	3	1646	0,16	0,4	0,44
376	4	7524	7676	0	1646	0	0,08	0,92
377	4	7524	7676	0	1646	0	0,08	0,92
378	4	7524	7676	0	1646	0	0,08	0,92
379	4	7524	7676	0	1646	0	0,08	0,92
380	3	9504	9696	3	1646	0,22	0,43	0,35
381	3	9504	9696	3	1646	0,22	0,43	0,35
382	3	9504	9696	3	1646	0,22	0,43	0,35
383	4	11088	11312	3	1646	0,22	0,43	0,35
384	2	6336	6464	3	1646	0,32	0,38	0,3
385	4	5940	6060	4	1646	0,33	0,3	0,37
386	4	12672	12928	4	1646	0,33	0,3	0,37
387	4	9504	9696	4	1646	0,33	0,3	0,37
388	4	9504	9696	4	1646	0,33	0,25	0,42
389	4	13167	13433	5	1646	0,42	0,3	0,28
390	4	13167	13433	5	1646	0,2	0,48	0,32
391	4	16929	17271	6	1646	0,2	0,48	0,32
392	4	16929	17271	6	1646	0,2	0,48	0,32
393	4	13167	13433	5	1646	0,44	0,28	0,28
394	4	14256	14544	6	1646	0,16	0,38	0,46
395	2	13167	13433	5	1646	0,45	0,29	0,26
396	2	13167	13433	5	1646	0,45	0,29	0,26
397	4	13167	13433	5	1646	0,33	0,33	0,34
398	4	13167	13433	5	1646	0,33	0,33	0,34
399	4	13167	13433	5	1646	0,33	0,33	0,34
400	4	13167	13433	5	1646	0,33	0,33	0,34

Table E.8: Real instances – part 8

no.	κ	$\underline{I}$	$\bar{I}$	$\lvert S\rvert$	$\lvert B\rvert$	Prob(low)	Prob(normal)	Prob(high)
401	4	13167	13433	5	1646	0,54	0,22	0,24
402	4	13167	13433	5	1646	0,54	0,22	0,24
403	4	13167	13433	5	1646	0,54	0,22	0,24
404	4	13167	13433	5	1646	0,47	0,26	0,27
405	4	13167	13433	5	1646	0,47	0,26	0,27
406	4	11286	11514	3	1646	0,22	0,43	0,35
407	4	9405	9595	3	1646	0,22	0,43	0,35
408	4	13167	13433	3	1646	0,22	0,43	0,35
409	4	13167	13433	3	1646	0,22	0,43	0,35
410	4	7524	7676	3	1646	0,22	0,43	0,35
411	4	8316	8484	3	1646	0,32	0,38	0,3
412	4	8316	8484	3	1646	0,32	0,38	0,3
413	4	12474	12726	3	1646	0,21	0,5	0,29
414	4	12474	12726	3	1646	0,21	0,5	0,29
415	4	12474	12726	4	1646	0,33	0,3	0,37
416	4	12474	12726	4	1646	0,33	0,3	0,37
417	4	12672	12928	4	1646	0,33	0,3	0,37
418	4	9504	9696	4	1646	0,33	0,3	0,37
419	4	10395	10605	3	1646	0,5	0,31	0,2
420	4	10395	10605	3	1646	0,5	0,31	0,2
421	4	9504	9696	4	1646	0,21	0,37	0,42
422	4	9504	9696	4	1646	0,21	0,37	0,42
423	4	9504	9696	4	1646	0,21	0,37	0,42
424	4	9504	9696	4	1646	0,21	0,37	0,42
425	4	11286	11514	4	1646	0,36	0,33	0,31
426	4	11286	11514	4	1646	0,33	0,25	0,42
427	4	11088	11312	5	1646	0,42	0,3	0,28
428	4	11088	11312	5	1646	0,42	0,3	0,28
429	3	11088	11312	5	1646	0,42	0,3	0,28
430	3	11088	11312	5	1646	0,42	0,3	0,28
431	3	11088	11312	5	1646	0,42	0,3	0,28
432	4	14256	14544	6	1646	0,47	0,29	0,24
433	4	14256	14544	6	1646	0,16	0,38	0,46
434	2	7920	8080	2	1646	0,22	0,43	0,35
435	4	11088	11312	3	1646	0,22	0,43	0,35
436	4	9504	9696	4	1646	0,21	0,37	0,42
437	4	9504	9696	4	1646	0,21	0,37	0,42
438	4	9504	9696	4	1646	0,34	0,38	0,29
439	4	9504	9696	4	1646	0,34	0,38	0,29
440	2	11088	11312	6	1646	0,34	0,38	0,29
441	3	9504	9696	4	1646	0,32	0,29	0,38
442	3	9504	9696	4	1646	0,32	0,29	0,38
443	4	14256	14544	6	1656	0,37	0,28	0,35
444	13	16929	17271	7	1646	0,16	0,38	0,46
445	4	16929	17271	6	1656	0,18	0,31	0,51
446	4	16929	17271	6	1656	0,16	0,38	0,46
447	4	16929	17271	6	1656	0,25	0,36	0,39
448	4	16929	17271	6	1656	0,16	0,38	0,46
449	4	16929	17271	6	1656	0,34	0,24	0,42
450	4	13167	13433	5	1656	0,47	0,29	0,24

Table E.9: Real instances – part 9

no.	κ	$\underline{I}$	$\bar{I}$	$\lvert S\rvert$	$\lvert B\rvert$	Prob(low)	Prob(normal)	Prob(high)
451	4	16929	17271	6	1656	0,16	0,38	0,46
452	4	16929	17271	7	1656	0,16	0,38	0,46
453	4	16929	17271	6	1656	0,16	0,38	0,46
454	4	13167	13433	5	1656	0,33	0,33	0,34
455	4	13167	13433	5	1656	0,33	0,33	0,34
456	4	13167	13433	5	1656	0,33	0,33	0,34
457	4	13167	13433	5	1656	0,33	0,33	0,34
458	2	13167	13433	5	1656	0,45	0,29	0,26
459	2	13167	13433	5	1656	0,45	0,29	0,26
460	4	13167	13433	5	1656	0,78	0,22	0
461	4	13167	13433	5	1656	0,54	0,22	0,24
462	4	13167	13433	5	1656	0,54	0,22	0,24
463	4	13167	13433	5	1656	0,54	0,22	0,24
464	4	13167	13433	5	1656	0,47	0,26	0,27
465	4	13167	13433	5	1656	0,47	0,26	0,27
466	4	13167	13433	5	1656	0,47	0,26	0,27
467	4	13167	13433	5	1656	0,47	0,26	0,27
468	4	11088	11312	5	1656	0,54	0,22	0,24
469	4	15840	16160	7	1656	0,49	0,32	0,19
470	4	15840	16160	7	1656	0,38	0,34	0,28
471	4	15840	16160	7	1656	0,38	0,34	0,28
472	4	15840	16160	7	1656	0,39	0,3	0,31
473	3	9504	9696	3	1656	0,22	0,43	0,35
474	4	11088	11312	3	1656	0,22	0,43	0,35
475	4	11088	11312	3	1656	0,22	0,43	0,35
476	4	11088	11312	3	1656	0,22	0,43	0,35
477	4	14256	14544	4	1656	0,33	0,3	0,37
478	4	9504	9696	4	1656	0,33	0,3	0,37
479	4	9504	9696	4	1656	0,33	0,3	0,37
480	4	9504	9696	4	1656	0,33	0,3	0,37
481	4	9504	9696	4	1656	0,33	0,3	0,37
482	4	9504	9696	4	1656	0,33	0,3	0,37
483	4	7920	8080	3	1656	0,5	0,31	0,2
484	4	11286	11514	4	1656	0,21	0,37	0,42
485	4	11286	11514	4	1656	0,21	0,37	0,42
486	4	11286	11514	4	1656	0,21	0,37	0,42
487	4	11286	11514	4	1656	0,21	0,37	0,42
488	4	10395	10605	4	1656	0,32	0,29	0,38
489	4	10395	10605	4	1656	0,32	0,29	0,38
490	4	10395	10605	4	1656	0,32	0,29	0,38
491	4	10395	10605	4	1656	0,32	0,29	0,38
492	4	9504	9696	4	1656	0,38	0,33	0,29
493	4	13167	13433	5	1656	0,33	0,34	0,33
494	4	13167	13433	5	1656	0,33	0,34	0,33
495	4	13167	13433	5	1656	0,42	0,3	0,28
496	4	13167	13433	5	1656	0,33	0,34	0,33
497	4	13167	13433	5	1656	0,33	0,34	0,33
498	4	16929	17271	6	1656	0,34	0,33	0,34
499	4	16929	17271	6	1656	0,34	0,33	0,34
500	4	16929	17271	6	1656	0,37	0,28	0,35

Table E.10: Real instances – part 10

no.	κ	$\underline{I}$	$\bar{I}$	$\lvert S\rvert$	$\lvert B\rvert$	Prob(low)	Prob(normal)	Prob(high)
501	4	16929	17271	6	1656	0,2	0,48	0,32
502	4	16929	17271	6	1656	0,34	0,33	0,34
503	4	16929	17271	6	1656	0,37	0,28	0,35
504	4	19305	19695	7	1656	0,39	0,3	0,31
505	4	19305	19695	7	1656	0,38	0,34	0,28
506	4	19305	19695	7	1656	0,49	0,32	0,19
507	4	19305	19695	7	1656	0,38	0,34	0,28
508	4	9504	9696	3	1656	0,22	0,43	0,35
509	4	9504	9696	3	1656	0,22	0,43	0,35
510	3	9504	9696	3	1656	0,22	0,43	0,35
511	4	12672	12928	4	1656	0,22	0,43	0,35
512	4	11088	11312	3	1656	0,22	0,43	0,35
513	3	6336	6464	3	1656	0,21	0,5	0,29
514	4	9504	9696	4	1656	0,42	0,3	0,28
515	4	9504	9696	4	1656	0,42	0,3	0,28
516	4	9504	9696	4	1656	0,42	0,3	0,28
517	4	13167	13433	5	1656	0,42	0,3	0,28
518	4	9504	9696	4	1656	0,33	0,34	0,33
519	4	13167	13433	5	1656	0,33	0,34	0,33
520	4	11088	11312	5	1656	0,33	0,34	0,33
521	4	16929	17271	6	1656	0,16	0,38	0,46
522	4	16929	17271	6	1656	0,16	0,38	0,46
523	4	16929	17271	6	1656	0,16	0,38	0,46
524	4	13167	13433	5	1656	0,47	0,29	0,24
525	4	16929	17271	6	1656	0,16	0,38	0,46
526	4	16929	17271	6	1656	0,16	0,38	0,46
527	4	16929	17271	6	1656	0,25	0,36	0,39
528	4	13167	13433	5	1656	0,33	0,33	0,34
529	4	13167	13433	5	1656	0,33	0,33	0,34
530	4	13167	13433	5	1656	0,33	0,33	0,34
531	4	13167	13433	5	1656	0,33	0,33	0,34
532	4	13167	13433	5	1656	0,33	0,33	0,34
533	4	13167	13433	5	1656	0,33	0,33	0,34
534	4	13167	13433	5	1656	0,33	0,33	0,34
535	4	13167	13433	5	1656	0,33	0,33	0,34
536	4	13167	13433	5	1656	0,33	0,33	0,34
537	2	13167	13433	5	1656	0,45	0,29	0,26
538	2	13167	13433	5	1656	0,45	0,29	0,26
539	2	13167	13433	5	1656	0,45	0,29	0,26
540	4	13167	13433	5	1656	0,54	0,22	0,24
541	4	13167	13433	5	1656	0,54	0,22	0,24
542	4	13167	13433	5	1656	0,54	0,22	0,24
543	4	13167	13433	5	1656	0,47	0,26	0,27
544	4	13167	13433	5	1656	0,47	0,26	0,27
545	4	9405	9595	3	1656	0,07	0,3	0,64
546	4	9405	9595	3	1656	0,16	0,4	0,44
547	4	9405	9595	4	1656	0,07	0,3	0,64
548	4	9405	9595	3	1656	0,07	0,3	0,64
549	4	9405	9595	3	1656	0,07	0,3	0,64
550	4	9405	9595	3	1656	0,07	0,3	0,64

Table E.11: Real instances – part 11

no.	κ	$\underline{I}$	$\bar{I}$	$\lvert S\rvert$	$\lvert B\rvert$	Prob(low)	Prob(normal)	Prob(high)
551	4	7920	8080	3	1656	0,22	0,43	0,35
552	4	9504	9696	3	1656	0,22	0,43	0,35
553	4	9504	9696	3	1656	0,22	0,43	0,35
554	4	9504	9696	3	1656	0,22	0,43	0,35
555	4	11088	11312	3	1656	0,22	0,43	0,35
556	4	11088	11312	3	1656	0,22	0,43	0,35
557	4	9504	9696	4	1656	0,33	0,3	0,37
558	4	12672	12928	4	1656	0,33	0,3	0,37
559	4	9504	9696	4	1656	0,33	0,3	0,37
560	4	13167	13433	5	1656	0,42	0,3	0,28
561	4	16929	17271	6	1656	0,2	0,48	0,32
562	4	16929	17271	6	1656	0,37	0,28	0,35
563	4	13167	13433	5	1656	0,2	0,44	0,36
564	4	16929	17271	6	1656	0,37	0,28	0,35
565	4	16929	17271	6	1656	0,37	0,28	0,35
566	4	16929	17271	6	1656	0,25	0,26	0,5
567	4	16929	17271	6	1656	0,37	0,28	0,35
568	4	16929	17271	6	1656	0,34	0,33	0,34
569	4	13167	13433	5	1656	0,2	0,48	0,32
570	4	16929	17271	6	1656	0,37	0,28	0,35
571	4	16929	17271	6	1656	0,37	0,28	0,35
572	4	16929	17271	6	1656	0,34	0,33	0,34
573	4	13167	13433	5	1656	0,1	0,43	0,48
574	4	15840	16160	7	1656	0,38	0,34	0,28
575	4	9405	9595	3	1656	0,22	0,43	0,35
576	4	9405	9595	3	1656	0,22	0,43	0,35
577	4	9306	9494	3	1656	0,22	0,43	0,35
578	4	9405	9595	2	1656	0,22	0,43	0,35
579	4	11088	11312	3	1656	0,22	0,43	0,35
580	4	11088	11312	3	1656	0,22	0,43	0,35
581	1	5643	5757	3	1656	0,32	0,38	0,3
582	4	13959	14241	3	1656	0,21	0,5	0,29
583	3	6336	6464	3	1656	0,21	0,5	0,29
584	3	6336	6464	3	1656	0,21	0,5	0,29
585	4	9405	9595	3	1656	0	0	1
586	4	11286	11514	4	1656	0,33	0,3	0,37
587	4	11286	11514	4	1656	0,33	0,3	0,37
588	4	9405	9595	4	1656	0,33	0,3	0,37
589	4	11286	11514	4	1656	0,33	0,3	0,37
590	4	12474	12726	4	1656	0,34	0,38	0,29
591	4	14553	14847	4	1656	0,34	0,38	0,29
592	4	14553	14847	4	1656	0,34	0,38	0,29
593	3	18612	18988	4	1656	0,21	0,37	0,42
594	3	18612	18988	4	1656	0,21	0,37	0,42
595	3	18612	18988	4	1656	0,21	0,37	0,42
596	4	11286	11514	4	1656	0,38	0,33	0,29
597	4	11286	11514	4	1656	0,33	0,25	0,42
598	4	11286	11514	4	1656	0,36	0,33	0,31
599	4	11286	11514	4	1656	0,38	0,33	0,29
600	4	14256	14544	6	1676	0,37	0,28	0,35

Table E.12: Real instances – part 12

no.	κ	$\underline{I}$	$\bar{I}$	$\|S\|$	$\|B\|$	Prob(low)	Prob(normal)	Prob(high)
601	4	16929	17271	6	1676	0,34	0,24	0,42
602	4	13167	13433	5	1676	0,47	0,29	0,24
603	4	16929	17271	6	1676	0,16	0,38	0,46
604	4	16929	17271	6	1676	0,16	0,38	0,46
605	4	13514	13786	5	1676	0,38	0,34	0,28
606	4	13514	13786	5	1676	0,49	0,32	0,19
607	4	13514	13786	5	1676	0,38	0,34	0,28
608	4	13167	13433	5	1676	0,49	0,32	0,19
609	4	9405	9595	3	1676	0,22	0,43	0,35
610	4	9306	9494	3	1676	0,22	0,43	0,35
611	4	11286	11514	3	1676	0,22	0,43	0,35
612	4	11088	11312	3	1676	0,22	0,43	0,35
613	4	11088	11312	3	1676	0,22	0,43	0,35
614	1	5643	5757	3	1676	0,32	0,38	0,3
615	4	9405	9595	3	1676	0,32	0,38	0,3
616	4	9405	9595	3	1676	0,32	0,38	0,3
617	3	9504	9696	4	1676	0,32	0,29	0,38
618	3	9504	9696	4	1676	0,32	0,29	0,38
619	3	11088	11312	5	1676	0,42	0,3	0,28
620	4	16929	17271	6	1676	0,2	0,44	0,36
621	4	13167	13433	5	1676	0,34	0,33	0,34
622	4	16929	17271	7	1676	0,2	0,48	0,32
623	4	16929	17271	6	1676	0,37	0,28	0,35
624	4	16929	17271	6	1676	0,37	0,28	0,35
625	4	13167	13433	5	1676	0,2	0,44	0,36
626	4	16929	17271	6	1676	0,37	0,28	0,35
627	4	16929	17271	6	1676	0,34	0,33	0,34
628	4	16929	17271	6	1676	0,2	0,48	0,32
629	4	16929	17271	6	1676	0,1	0,43	0,48
630	2	13167	13433	5	1676	0,45	0,29	0,26
631	4	13167	13433	5	1676	0,54	0,22	0,24
632	4	13167	13433	5	1676	0,54	0,22	0,24
633	4	13167	13433	5	1676	0,54	0,22	0,24
634	4	13167	13433	5	1676	0,54	0,22	0,24
635	4	13167	13433	5	1676	0,47	0,26	0,27
636	4	13167	13433	5	1676	0,47	0,26	0,27
637	4	13167	13433	5	1676	0,47	0,26	0,27
638	4	11088	11312	5	1676	0,54	0,22	0,24
639	4	9405	9595	3	1676	0,07	0,3	0,64
640	4	9405	9595	3	1676	0,07	0,3	0,64
641	4	9405	9595	3	1676	0,07	0,3	0,64
642	4	9405	9595	3	1676	0,07	0,3	0,64
643	4	9405	9595	3	1676	0,16	0,4	0,44
644	4	9405	9595	3	1676	0,16	0,4	0,44
645	4	11088	11312	3	1676	0,22	0,43	0,35
646	4	11088	11312	3	1676	0,22	0,43	0,35
647	4	9504	9696	3	1676	0,22	0,43	0,35
648	4	9504	9696	3	1676	0,22	0,43	0,35
649	4	11088	11312	3	1676	0,22	0,43	0,35
650	4	7920	8080	3	1676	0,22	0,43	0,35

Table E.13: Real instances – part 13

no.	κ	$\underline{I}$	$\bar{I}$	$\|S\|$	$\|B\|$	Prob(low)	Prob(normal)	Prob(high)
651	4	9504	9696	4	1676	0,33	0,3	0,37
652	4	12672	12928	4	1676	0,33	0,3	0,37
653	4	9504	9696	4	1676	0,33	0,3	0,37
654	4	9504	9696	4	1676	0,33	0,3	0,37
655	4	9504	9696	4	1676	0,33	0,3	0,37
656	4	9504	9696	4	1676	0,33	0,3	0,37
657	3	11088	11312	5	1676	0,34	0,38	0,29
658	4	9504	9696	4	1676	0,21	0,37	0,42
659	4	9504	9696	4	1676	0,21	0,37	0,42
660	2	9504	9696	5	1676	0,34	0,38	0,29
661	2	9504	9696	5	1676	0,34	0,38	0,29
662	4	9504	9696	4	1676	0,34	0,38	0,29
663	4	11286	11514	4	1676	0,36	0,33	0,31
664	4	11286	11514	4	1676	0,33	0,25	0,42
665	4	11286	11514	4	1676	0,38	0,33	0,29
666	4	11286	11514	4	1676	0,36	0,33	0,31
667	2	9504	9696	5	1676	0,38	0,33	0,29
668	4	11088	11312	5	1676	0,33	0,34	0,33
669	4	13167	13433	5	1676	0,47	0,29	0,24
670	4	16929	17271	6	1676	0,16	0,38	0,46
671	4	16929	17271	6	1676	0,16	0,38	0,46
672	4	19305	19695	7	1676	0,38	0,34	0,28
673	4	19305	19695	7	1676	0,39	0,3	0,31
674	4	19305	19695	7	1676	0,38	0,34	0,28
675	4	19305	19695	7	1676	0,49	0,32	0,19
676	4	19305	19695	7	1676	0,39	0,3	0,31
677	4	19305	19695	7	1676	0,38	0,34	0,28
678	4	19305	19695	7	1676	0,49	0,32	0,19
679	4	19305	19695	7	1676	0,38	0,34	0,28
680	4	19305	19695	7	1676	0,38	0,34	0,28
681	4	19305	19695	7	1676	0,34	0,32	0,34
682	4	19305	19695	7	1676	0,34	0,32	0,34
683	4	11088	11312	3	1676	0,22	0,43	0,35
684	4	11088	11312	3	1676	0,22	0,43	0,35
685	4	11088	11312	3	1676	0,22	0,43	0,35
686	4	9504	9696	3	1676	0,5	0,31	0,2
687	2	9504	9696	5	1676	0,38	0,33	0,29
688	4	13167	13433	5	1676	0,33	0,34	0,33
689	4	13167	13433	5	1676	0,33	0,34	0,33
690	4	13167	13433	5	1676	0,33	0,34	0,33
691	4	13167	13433	5	1676	0,33	0,34	0,33
692	4	13167	13433	5	1676	0,33	0,34	0,33
693	4	13167	13433	5	1676	0,42	0,3	0,28
694	4	13167	13433	5	1676	0,35	0,39	0,26
695	4	11187	11413	5	1676	0,42	0,3	0,28
696	4	16929	17271	6	1676	0,34	0,33	0,34
697	4	13167	13433	5	1676	0,1	0,43	0,48
698	4	16929	17271	6	1676	0,2	0,44	0,36
699	4	16929	17271	6	1676	0,34	0,33	0,34
700	4	16929	17271	6	1676	0,34	0,33	0,34

Table E.14: Real instances – part 14

no.	κ	$\underline{I}$	$\bar{I}$	$\|S\|$	$\|B\|$	Prob(low)	Prob(normal)	Prob(high)
701	4	13167	13433	5	1676	0,2	0,44	0,36
702	4	16929	17271	6	1676	0,2	0,44	0,36
703	4	13167	13433	5	1686	0,54	0,22	0,24
704	4	13167	13433	5	1676	0,38	0,34	0,28
705	4	13167	13433	5	1676	0,39	0,3	0,31
706	4	13167	13433	5	1676	0,38	0,34	0,28
707	4	13514	13786	7	1676	0,49	0,32	0,19
708	4	13514	13786	7	1676	0,39	0,3	0,31
709	4	13167	13433	5	1676	0,38	0,34	0,28
710	4	13514	13786	7	1676	0,38	0,34	0,28
711	4	10395	10605	3	1676	0,22	0,43	0,35
712	4	7920	8080	2	1676	0,22	0,43	0,35
713	2	6336	6464	3	1676	0,22	0,43	0,35
714	4	9504	9696	3	1676	0,22	0,43	0,35
715	4	11088	11312	3	1676	0,22	0,43	0,35
716	4	9405	9595	3	1676	0,32	0,38	0,3
717	4	6237	6363	6	1676	0,32	0,38	0,3
718	1	6237	6363	3	1676	0,32	0,38	0,3
719	2	6237	6363	3	1676	0,32	0,38	0,3
720	2	6237	6363	3	1676	0,32	0,38	0,3
721	4	13959	14241	3	1676	0,21	0,5	0,29
722	4	13959	14241	3	1676	0,21	0,5	0,29
723	4	12474	12726	4	1676	0,33	0,3	0,37
724	4	10395	10605	4	1676	0,33	0,3	0,37
725	4	9306	9494	3	1676	0,5	0,31	0,2
726	4	9306	9494	3	1676	0,5	0,31	0,2
727	4	12474	12726	4	1676	0,34	0,38	0,29
728	4	12474	12726	4	1676	0,34	0,38	0,29
729	4	12474	12726	4	1676	0,34	0,38	0,29
730	4	12474	12726	4	1676	0,21	0,37	0,42
731	4	12474	12726	4	1676	0,21	0,37	0,42
732	4	12474	12726	4	1676	0,21	0,37	0,42
733	4	14553	14847	4	1676	0,32	0,29	0,38
734	4	14553	14847	4	1676	0,32	0,29	0,38
735	4	14553	14847	4	1676	0,32	0,29	0,38
736	4	11286	11514	4	1676	0,38	0,33	0,29
737	4	11286	11514	4	1676	0,33	0,25	0,42
738	4	11286	11514	4	1676	0,38	0,33	0,29
739	4	11286	11514	4	1676	0,38	0,33	0,29
740	4	11286	11514	4	1676	0,38	0,33	0,29
741	4	14256	14544	6	1676	0,37	0,28	0,35
742	4	14256	14544	6	1676	0,37	0,28	0,35
743	4	13167	13433	5	1676	0	0,13	0,87
744	4	16929	17271	6	1676	0,16	0,38	0,46
745	4	16929	17271	6	1676	0,16	0,38	0,46
746	4	13167	13433	5	1676	0	0,13	0,87
747	4	16929	17271	6	1676	0,16	0,38	0,46
748	4	19305	19695	7	1676	0,39	0,3	0,31
749	4	19305	19695	7	1676	0,49	0,32	0,19
750	4	19305	19695	7	1676	0,38	0,34	0,28

Table E.15: Real instances – part 15

no.	κ	$\underline{I}$	$\bar{I}$	$\|S\|$	$\|B\|$	Prob(low)	Prob(normal)	Prob(high)
751	4	19305	19695	7	1676	0,38	0,34	0,28
752	4	19305	19695	7	1676	0,39	0,3	0,31
753	4	19305	19695	7	1676	0,38	0,34	0,28
754	4	19305	19695	7	1676	0,38	0,34	0,28
755	4	19305	19695	7	1676	0,49	0,32	0,19
756	4	19305	19695	7	1676	0,49	0,32	0,19
757	4	7524	7676	3	1676	0,07	0,3	0,64
758	4	9405	9595	3	1676	0,07	0,3	0,64
759	4	9405	9595	3	1676	0,07	0,3	0,64
760	4	9405	9595	3	1676	0,07	0,3	0,64
761	4	9405	9595	3	1676	0,16	0,4	0,44
762	4	9405	9595	3	1676	0,16	0,4	0,44
763	4	9405	9595	3	1676	0,07	0,3	0,64
764	4	9405	9595	3	1676	0,07	0,3	0,64
765	4	9405	9595	3	1676	0,16	0,4	0,44
766	4	9405	9595	3	1676	0,16	0,4	0,44
767	4	9405	9595	3	1676	0,07	0,3	0,64
768	4	9405	9595	3	1676	0,22	0,43	0,35
769	4	9405	9595	3	1676	0,22	0,43	0,35
770	4	9405	9595	3	1676	0,32	0,38	0,3
771	4	14553	14847	4	1676	0,21	0,37	0,42
772	4	14553	14847	4	1676	0,21	0,37	0,42
773	4	12474	12726	4	1676	0,32	0,29	0,38
774	4	12474	12726	4	1676	0,32	0,29	0,38
775	4	9504	9696	4	1676	0,33	0,25	0,42
776	4	9504	9696	4	1676	0,36	0,33	0,31
777	4	9504	9696	4	1676	0,33	0,25	0,42
778	4	11286	11514	4	1676	0,42	0,3	0,28
779	4	11286	11514	4	1676	0,33	0,34	0,33
780	4	11286	11514	4	1676	0,33	0,34	0,33
781	4	16929	17271	6	1686	0,34	0,33	0,34
782	4	13167	13433	5	1686	0,2	0,48	0,32
783	4	13167	13433	5	1686	0,2	0,44	0,36
784	4	16929	17271	6	1686	0,37	0,28	0,35
785	4	16929	17271	6	1686	0,37	0,28	0,35
786	4	16929	17271	6	1686	0,1	0,43	0,48
787	4	13167	13433	5	1686	0,2	0,44	0,36
788	4	16929	17271	6	1686	0,37	0,28	0,35
789	4	13167	13433	5	1686	0,47	0,26	0,27
790	4	11286	11514	3	1686	0,22	0,43	0,35
791	4	9306	9494	3	1686	0,22	0,43	0,35
792	4	11286	11514	3	1686	0,22	0,43	0,35
793	4	11286	11514	3	1686	0,22	0,43	0,35
794	4	11088	11312	3	1686	0,22	0,43	0,35
795	3	7524	7676	3	1686	0,32	0,38	0,3
796	3	6336	6464	3	1686	0,21	0,5	0,29
797	4	9504	9696	4	1686	0,33	0,3	0,37
798	4	12672	12928	4	1686	0,33	0,3	0,37
799	3	9504	9696	4	1686	0,33	0,3	0,37
800	3	9504	9696	4	1686	0,33	0,3	0,37

Table E.16: Real instances – part 16

no.	κ	$\underline{I}$	$\bar{I}$	$\|S\|$	$\|B\|$	Prob(low)	Prob(normal)	Prob(high)
801	4	9504	9696	4	1686	0,33	0,3	0,37
802	4	9504	9696	4	1686	0,33	0,3	0,37
803	4	12672	12928	4	1686	0,33	0,3	0,37
804	4	9504	9696	4	1686	0,21	0,37	0,42
805	4	9504	9696	4	1686	0,21	0,37	0,42
806	4	9504	9696	4	1686	0,34	0,38	0,29
807	4	9504	9696	4	1686	0,34	0,38	0,29
808	4	11286	11514	4	1686	0,33	0,25	0,42
809	4	11286	11514	4	1686	0,36	0,33	0,31
810	4	11286	11514	4	1686	0,38	0,33	0,29
811	4	11286	11514	4	1686	0,38	0,33	0,29
812	4	13167	13433	5	1686	0,2	0,48	0,32
813	4	16929	17271	6	1686	0,37	0,28	0,35
814	4	16929	17271	6	1686	0,37	0,28	0,35
815	4	16929	17271	6	1686	0,37	0,28	0,35
816	4	16929	17271	6	1686	0,16	0,38	0,46
817	4	16929	17271	6	1686	0,16	0,38	0,46
818	4	16929	17271	6	1686	0,16	0,38	0,46
819	4	16929	17271	6	1686	0,16	0,38	0,46
820	4	16929	17271	6	1686	0,16	0,38	0,46
821	4	13167	13433	5	1686	0,33	0,33	0,34
822	4	13167	13433	5	1686	0,33	0,33	0,34
823	2	13167	13433	5	1686	0,45	0,29	0,26
824	2	13167	13433	5	1686	0,45	0,29	0,26
825	4	13167	13433	5	1686	0,54	0,22	0,24
826	4	13167	13433	5	1686	0,54	0,22	0,24
827	4	13167	13433	5	1686	0,47	0,26	0,27
828	4	13167	13433	5	1686	0,47	0,26	0,27
829	4	13167	13433	5	1686	0,47	0,26	0,27
830	4	13167	13433	5	1686	0,47	0,26	0,27
831	4	14207	14493	5	1686	0,38	0,34	0,28
832	4	13167	13433	5	1686	0,42	0,3	0,28
833	4	13167	13433	5	1686	0,33	0,34	0,33
834	4	13167	13433	5	1686	0,33	0,34	0,33
835	4	13167	13433	5	1686	0,33	0,34	0,33
836	4	13167	13433	5	1686	0,2	0,48	0,32
837	4	16929	17271	6	1686	0,37	0,28	0,35
838	4	13167	13433	5	1686	0,37	0,28	0,35
839	4	16929	17271	6	1686	0,2	0,44	0,36
840	4	16929	17271	6	1686	0,37	0,28	0,35
841	4	18810	19190	6	1686	0,16	0,38	0,46
842	4	11286	11514	4	1686	0,22	0,43	0,35
843	4	11286	11514	4	1686	0,22	0,43	0,35
844	4	15048	15352	3	1686	0,22	0,43	0,35
845	4	16286	16614	3	1686	0,22	0,43	0,35
846	3	7920	8080	2	1686	0,22	0,43	0,35
847	4	11286	11514	3	1686	0,22	0,43	0,35
848	2	6237	6363	3	1686	0,32	0,38	0,3
849	2	6336	6464	3	1686	0,32	0,38	0,3
850	3	6237	6363	3	1686	0,21	0,5	0,29
851	2	5643	5757	3	1686	0,21	0,5	0,29
852	4	11286	11514	4	1686	0,33	0,3	0,37
853	4	11286	11514	4	1686	0,33	0,3	0,37
854	4	11286	11514	4	1686	0,33	0,3	0,37
855	4	12474	12726	3	1686	0,5	0,31	0,2
856	4	12474	12726	3	1686	0,5	0,31	0,2
857	4	9504	9696	4	1686	0,34	0,38	0,29
858	4	9504	9696	4	1686	0,34	0,38	0,29
859	4	9504	9696	4	1686	0,21	0,37	0,42
860	4	13167	13433	5	1686	0,42	0,3	0,28

Table E.17: Real instances – part 18

List of Figures

List of Tables

List of Algorithms

Bibliography

[ABG10] O. Aalen, O. Borgan, and H. Gjessing. *Survival and Event History Analysis: A Process Point of View (Statistics for Biology and Health)*. Springer, 2010.

[ADG98] R. Anupindi, M. Dada, and S. Gupta. Estimation of consumer demand with stock-out based substitution: An application to vending machine products. *Marketing Science*, 17(4):406–423, 1998.

[ADSZ10] M. Avriel, E. Diewert, S. Schaible, and I. Zang. *Generalized Concavity (Classics in Applied Mathematics)*. Society for Industrial & Applied Mathematics, 2010.

[ALMP05] R. Andrade, A. Lisser, N. Maculan, and G. Plateau. B&B frameworks for the capacity expansion of high speed telecommunication networks under uncertainty. *Annals of Operations Research*, 140:49–65, 2005.

[ANW06] K. Aardal, George L. Nemhauser, and R. Weismantel, editors. *Handbooks in Operations Research and Management Science, Volume 12: Discrete Optimization*. North Holland, 2006.

[AP06] E. Adida and G. Perakis. A robust optimization approach to dynamic pricing and inventory control with no backorders. *Mathematical Programming*, 107:97–129, 2006.

[AP10] E. Adida and G. Perakis. Dynamic pricing and inventory control: Uncertainty and competition. *Operations Research*, pages 289–302, 2010.

[AS03] N. Agrawal and S. Smith. Optimal prepack design in retail supply chains. INFORMS Conference Presentation, Atlanta, GA, 2003.

[ASvdVF06] M. Albareda-Sambola, M. H. van der Vlerk, and E. Fernández. Exact solutions to a class of stochastic generalized assignment problems. *European Journal of Operations Research*, 173(2):465–487, 2006.

[ATS04] S. Ahmed, M. Tawarmalani, and N. V. Sahinidis. A finite branch-and-bound algorithm for two-stage stochastic integer programs. 2004.

[Bak10] M. T. Bakema. Using pre-packs to preload stores at the beginning of the high-demand season. Master's thesis, Technische Universiteit Eindhoven, 2010.

[BC03] G. Bitran and R. Caldentey. An overview of pricing models for revenue management. *Manufacturing & Service Operations Management*, 5(3):203–229, 2003.

[BCJ+10] I. Bomze, M. Chimani, M. Jünger, I. Ljubić, P. Mutzel, and B. Zey. Solving two-stage stochastic steiner tree problems by two-stage branch-and-cut. Lecture Notes in Computer Science 6506, 2010.

[BdB05] D. Bertsimas and S. de Boer. Dynamic pricing and inventory control for multiple products. *Journal of Revenue & Pricing Management*, 3:303–319, 2005.

[Bel10] R. Bellman. *Dynamic Programming (Princeton Landmarks in Mathematics)*. Princeton University Press, 2010.

[Ben62] J. F. Benders. Partitioning procedures for solving mixed variables programming problems. *Numerische Mathematik*, 4:238–252, 1962.

[Ber05] D. P. Bertsekas. *Dynamic Programming & Optimal Control, Vol. I*. Athena Scientific, 3rd edition, 2005.

[Bir82] J. R. Birge. The value of the stochastic solution in stochastic linear programs with fixed recourse. *Mathematical Programming*, 24:314–325, 1982.

[BJN+98] C. Barnhart, E. L. Johnson, G. L. Nemhauser, M. W. P. Savelsbergh, and P. H. Vance. Branch-and-price: Column generation for solving huge integer programs. *Operations Research*, 46(3):316–329, 1998.

[BL97] J. R. Birge and F. Louveaux. *Introduction to Stochastic Programming (Springer Series in Operations Research and Financial Engineering)*. Springer, corrected edition, 1997.

[Bre96] R. Breen. *Regression Models: Censored, Sample Selected, or Truncated Data (Quantitative Applications in the Social Sciences)*. Sage Publications, Inc, 1 edition, 1996.

[CCMGB10] A. J. Conejo, E. Castillo, R. Minguez, and R. Garcia-Bertrand. *Decomposition Techniques in Mathematical Programming: Engineering and Science Applications*. Springer, 2010.

[CGR93] B. Cherkassky, A. V. Goldberg, and T. Radzik. Shortest paths algorithms: Theory and experimental evaluation. *Mathematical Programming*, 73:129–174, 1993.

[CSSLS04] L. M. A. Chan, Z. J. M. Shen, D. Simchi-Levi, and J. Swann. Coordination of pricing and inventory decisions: A survey and classification. pages 335–392, 2004.

[CT98] C. Carøe and J. Tind. L-shaped decomposition of two-stage stochastic programs with integer recourse. *Mathematical Programming*, 83:451–464, 1998. 10.1007/BF02680570.

[DLR77] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the royal statistical society, Series B*, 39(1):1–38, 1977.

[DW60] G. B. Dantzig and P. Wolfe. Decomposition principle for linear programs. *Operations Research*, 8:101–111, 1960.

[EGK08] W. Elmaghraby, A. Gülcü, and P. Keskinocak. Designing optimal pre-announced markdowns in the presence of rational customers with multi-unit demands. *Manufacturing & Service Operations Management (MSOM)*, 10(1):126–148, 2008.

[FGT10] M. Freimer, L. Gao, and D. Thomas. Optimal inventory control with retail pre-packs. Technical report, The Pennsylvania State University, 2010.

[FH99] A. Federgruen and A. Heching. Combined pricing and inventory control under uncertainty. *Operations Research*, 47(3):454–475, 1999.

[FPP07] D. Freedman, R. Pisani, and R. Purves. *Statistics, 4th Edition*. W. W. Norton & Company, 4th edition, 2007.

[Fre09] Free Software Foundation's GNU General Public License. The r project for statistical computing, 2009.

[FT88] M. Fischetti and P. Toth. A new dominance procedure for combinatorial optimization problems. *Operations Research Letters*, 7(4):181–187, 1988.

[GJ79] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness (Series of Books in the Mathematical Sciences)*. W. H. Freeman, first edition edition, 1979.

[GKR09] C. Gaul, S. Kurz, and J. Rambau. The Combinatorics of (S,M,L,XL) or the best fitting delivery of T-shirts. In *Models and Algorithms for Optimization in Logistics*, Dagstuhl Seminar Proceedings, 2009.

[GKR10] C. Gaul, S. Kurz, and J. Rambau. On the lot-type design problem. *Optimization Methods and Software*, 25(2):217–227, 2010.

[GPM89] Carlos E. García, David M. Prett, and Manfred Morari. Model predictive control: Theory and practice—a survey. *Automatica*, 25(3):335 – 348, 1989.

[Gre03] W. H. Greene. *Econometric Analysis*. Prentice Hall, Upper Saddle River, NJ, 5. edition, 2003.

[GvR94] G. Gallego and G. van Ryzin. Optimal dynamic pricing of inventories with stochastic demand over finite horizons. *Management Science*, 40:999–1020, 1994.

[GvR97] G. Gallego and G. van Ryzin. A multiproduct dynamic pricing problem and its applications to network yield management. *Operations Research*, 45(1):24–41, 1997.

[Har10] F. E. Harrell. *Regression Modeling Strategies: With Applications to Linear Models, Logistic Regression, and Survival Analysis (Springer Series in Statistics)*. Springer, 1st ed. 2001 edition, 2010.

[HCL11] G. Heilporn, J. Cordeau, and G. Laporte. An integer l-shaped algorithm for the dial-a-ride problem with stochastic customer delays. *Discrete Applied Mathematics*, 159(9):883–895, 2011.

[HK62] M. Held and R. M. Karp. A dynamic programming approach to sequencing problems. *Journal of the Society of Industrial and Applied Mathematics*, 10:196–210, 1962.

[HLRO11] W. T. Huh, R. Levi, P. Rusmevichientong, and J. B. Orlin. Adaptive data-driven inventory control with censored demand based on kaplan-meier estimator. *Operations Research*, 59(4):929–941, 2011.

[HZ80] G. Y. Handler and I. Zang. A dual algorithm for the constrained shortest path problem. *Networks*, 10(4):293–309, 1980.

[Iba77] T. Ibaraki. The power of dominance relations in branch-and-bound algorithms. *Journal of the Association for Computing Machinery*, 24:264–279, 1977.

[ID05] S. Irnich and G. Desaulniers. Shortest path problems with resource constraints. Desaulniers, Guy (ed.) et al., Column generation. New York, NY: Springer. GERAD 25th Anniversary Series 5, 2005.

[JC11] A. Jouglet and J. Carlier. Dominance rules in combinatorial optimization problems. *European Journal of Operations Research*, 212(3):433–444, 2011.

[Kan06] G. K. Kanji. *100 Statistical Tests*. Sage Publications Ltd, 3rd edition, 2006.

[KKP02] D.G. Kleinbaum, M. Klein, and E.R. Pryor. *Logistic Regression: A Self-Learning Text*. Statistics for Biology and Health. Springer, 2002.

[KKR11a] M. Kießling, S. Kurz, and J. Rambau. An exact column-generation approach for the lot-type design problem. 2011. submitted to ISCO 2012.

[KKR11b] M. Kießling, S. Kurz, and J. Rambau. The integrated size and price optimization problem. 2011. accepted by Journal of Numerical Algebra, Control, and Optimization.

[KKR12] M. Kießling, T. Kreisel S. Kurz, and J. Rambau. Evaluation of a new supply strategy based on stochastic programming for a fashion discounter. 2012. submitted to Operations Research Proceedings 2012.

[KRSW08] S. Kurz, J. Rambau, J. Schlüchtermann, and R. Wolf. The top-dog index: A new measurement for the demand consistency of the size distribution in pre-pack orders for a fashion discounter with many small branches. 2008.

[KW94] P. Kall and S.W. Wallace. *Stochastic programming*. Wiley-Interscience series in systems and optimization. Wiley, 1994.

[LD05] M. E. Lübbecke and J. Desrosiers. Selected topics in column generation. *Operations Research*, 53(6):1007–1023, 2005.

[LD11] M. E. Lübbecke and J. Desrosiers. *Branch-Price-and-Cut Algorithms*. John Wiley & Sons, Inc., 2011.

[LL93] G. Laporte and F. V. Louveaux. The integer l-shaped method for stochastic integer programs with complete recourse. *Operations Research Letters*, 13(3):133–142, 1993.

[Lüb10] M. E. Lübbecke. *Column Generation*. John Wiley & Sons, Inc., 2010.

[Man58] A. Manne. Programming of economic lot sizes. *Management Science*, 4 (2):115–135, 1958.

[McC80] P. McCullagh. Regression models for ordinal data. *Journal of the Royal Statistical Society. Series B (*, 42:109–142, 1980.

[Mie99] K. M. Miettinen. *Evolutionary Algorithms in Engineering & Computer Science*. John Wiley and sons LTD, 1999.

[MM06] C. Maglaras and J. Meissner. Dynamic pricing strategies for multiproduct revenue management problems. *Manufacturing & Service Operations Management (MSOM)*, 8(2):136–148, 2006.

[Mon05] A. L. Montgomery. The implementation challenge of pricing decision support systems for retail managers. *Applied Stochastic Models in Business and Industry*, 21(4-5):367–378, 2005.

[Mon10] D. C. Montgomery. Applied statistics and probability for engineers, 5th edition, 2010.

[MZ12] A. J. Mersereau and D. Zhang. Markdown pricing with unknown fraction of strategic customers. *Manufacturing & Service Operations Management*, 14(3):355–370, 2012.

[Net06] S. Netessine. Dynamic pricing of inventory/capacity with infrequent price changes. *European Journal of Operations Research*, 174(1):553–580, 2006.

[Pré95] A. Prékopa. *Stochastic Programming*. Mathematics and Its Applications. Kluwer Academic Publishers, 1995.

[Raj06] V. Rajagopalan. *Selected Statistical Tests*. New Age International Pvt Ltd Publishers, 2006.

[Rus06] A. Ruszczynski. *Nonlinear Optimization*. Princeton University Press, 2006.

[Rya08] T. P. Ryan. *Modern Regression Methods (Wiley Series in Probability and Statistics)*. Wiley-Interscience, 2 edition, 2008.

[SA98] S. A. Smith and D. D. Achabal. Clearance pricing and inventory policies for retail chains. *Management Science*, 44(3):285–300, 1998.

[Sch98] A. Schrijver. *Theory of Linear and Integer Programming*. Wiley, 1998.

[Sch03] R. Schultz. Stochastic programming with integer variables. 2003.

[Sch11] K. Schade. *Stochastische Optimierung: Bestandsoptimierung in Mehrstufigen Lagernetzwerken*. Stochastic Programming. Vieweg+Teubner Verlag, 2011.

[SKS12] B. Sandikçi, N. Kong, and A. J. Schaefer. A hierarchy of bounds for stochastic mixed-integer programs. *Mathematical Programming, Series A*, 2012.

[Som62] R. H. Somers. A new asymmetric measure of association for ordinal variables. *American Sociological Review*, 27(6):799–811, 1962.

[SW06] E. F. Silva and R. K. Wood. Solving a class of stochastic mixed-integer programs with branch and price. 2006.

[SZ06] H. D. Sherali and X. Zhu. On solving discrete two-stage stochastic programs having mixed-integer first- and second-stage variables. *Mathematical Programming*, 108(2):597–616, 2006.

[SZ09] H. D. Sherali and X. Zhu. Two-stage stochastic mixed-integer programs: algorithms and insights. Gao, David Y. (ed.) et al., Advances in applied mathematics and global optimization. New York, NY: Springer. Advances in Mechanics and Mathematics 17, 2009.

[The57] H. Theil. A note on certainty equivalence in dynamic planning. *Econometrica*, 25(2):346–349, 1957.

[Tob58] J. Tobin. Estimation of Relationships for Limited Dependent Variables. *Econometrica*, 26(1):24–36, 1958.

[TOH04] S.Y. Teng, H.L. Ong, and H.C. Huang. An integer l-shaped algorithm for time-constrained traveling salesman problem with stochastic travel and service times. *Asia-Pacific Journal of Operational Research*, 21(2):241–257, 2004.

[vA03] L. von Auer. *Ökonometrie: Eine Einführung*. Springer Berlin, 2003.

[Van07] R. Vanderbei. *Linear Programming: Foundations and Extensions*. International Series in Operations Research & Management Science. Springer, 2007.

[Van11] F. Vanderbeck. Branching in branch-and-price: a generic scheme. *Mathematical Programming, Series A*, 130(2):249–294, 2011.

[vdV07] P. van der Vlist. *Synchronizing the Retail Supply Chain*. PhD thesis, Erasmus Universiteit Rotterdam, 2007.

[VvRR12] G. Vulcano, G.J. van Ryzin, and R. Ratliff. Estimating primary demand for substituable products from sales transaction data. *Operations Research*, 60(2):313–334, 2012.

[vZvDvW+09] S. van Zelst, K. van Donselaar, T. van Woensel, R. Broekmeulen, and J. Fransoo. Logistics drivers for shelf stacking in grocery retail stores: Potential for efficiency improvement. *International Journal of Production Economics*, 121(2):620–632, 2009.

[Wal43] A. Wald. Tests of Statistical Hypotheses Concerning Several Parameters When the Number of Observations is Large. *Transactions of the American Mathematical Society*, 54(3):426–482, 1943.

[Wil45] F. Wilcoxon. Individual Comparisons by Ranking Methods. *Biometrics Bulletin*, 1(6):80–83, 1945.

[Wol80] R. D. Wollmer. Two stage linear programming under uncertainty with 0–1 integer first stage variables. *Mathematical Programming*, 19:279–288, 1980.

[Wol98] L. A. Wolsey. *Integer Programming*. Wiley-Interscience, 1 edition, 1998.

[YAPT09] R. Yin, Y. Aviv, A. Pazgal, and C. S. Tang. Optimal markdown pricing: Implications of inventory display formats in the presence of strategic customers. *Management Science*, 55:1391–1408, 2009.

[ZZ00] W. Zhao and Y.-S. Zheng. Optimal dynamic pricing for perishable assets with nonhomogeneous demand. *Management Science*, 46(3):375–388, 2000.

Eidesstattliche Erklärung

Hiermit versichere ich, die vorliegende Arbeit selbständig angefertigt und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet zu haben. Ferner erkläre ich, dass ich nicht anderweitig mit oder ohne Erfolg versucht habe, diese oder eine andere Dissertation einzureichen. Ich habe keine gleichartige Doktorprüfung an einer anderen Hochschule endgültig nicht bestanden. Ich habe keine Hilfe von gewerblichen Promotionsberatern und -vermittlern oder ähnlichen Dienstleistern weder bisher in Anspruch genommen noch werde ich sie künftig in Anspruch nehmen.

Bayreuth, den 25.9.2012

Miriam Kießling