La collana della Scuola di Alta Formazione Dottorale accoglie le migliori tesi di dottorato dell'Università degli Studi di Bergamo, insignite della dignità di stampa e sottoposte a procedura di *blind peer review*.

Coping with uncertainty in logistic and production problems is the focus of this research, which addresses four problems characterized by the presence of uncertainty. The first two problems analyze a distribution system with uncertain demand in which transshipment and backordering are allowed, the third problem studies the allocation and rebalancing activities in a bikesharing system under uncertain bike demand, and the fourth problem deals with workforce planning decisions considering workers' stochastic learning curves. For all these applications, stochastic programming formulations are proposed and the importance of considering uncertainty explicitly in the models is assessed.

Rossana Cavagnini obtained her Ph.D. in "Analytics for Economics and Business" (XXXI cycle) from the University of Bergamo. Her passion for operations research has led her to develop research in stochastic programming with applications in transportation, logistics and production management. She has conducted part of her research at the Loyola University in Chicago (U.S.A) and at CIRRELT in Montréal (Canada). Currently, she is postdoc at the Deutsche Post Chair – Optimization of Distribution Networks at RWTH University, Germany.

ISBN: 978-88-97413-41-7
DOI: 10.6092/978-88-97413-41-7

Rossana Cavagnini STOCHASTIC PROGRAMMING MODELS

24

Collana della Scuola di Alta Formazione Dottorale
- 24 -

Rossana Cavagnini

STOCHASTIC PROGRAMMING MODELS
Applications to logistics, bikesharing and production management

UNIVERSITÀ DEGLI STUDI DI BERGAMO

2021

Collana della Scuola di Alta Formazione Dottorale

Diretta da Paolo Cesaretti

Ogni volume è sottoposto a *blind peer review*.

ISSN: 2611-9927

Sito web: https://aisberg.unibg.it/handle/10446/130100

Rossana Cavagnini

STOCHASTIC PROGRAMMING MODELS
Applications to logistics, bikesharing and production management

Università degli Studi di Bergamo

2021

Stochastic programming models. Applications to logistics, bikesharing and production management /
Rossana Cavagnini. – Bergamo :
Università degli Studi di Bergamo, 2021.
(Collana della Scuola di Alta Formazione Dottorale; 24)

ISBN: 978-88-97413-41-7
DOI: 10.6092/978-88-97413-41-7

Progetto grafico: Servizi Editoriali – Università degli Studi di Bergamo
© 2018 Università degli Studi di Bergamo
via Salvecchio, 19
24129 Bergamo
Cod. Fiscale 80004350163
P. IVA 01612800167

https://aisberg.unibg.it/handle/10446/175490

To my parents and to Michele.

Acknowledgements

I sincerely thank my supervisors, Prof. F. Maggioni and Prof. L. Bertazzi, for the passion for Operations Research shared with me, for their patience and for all their precious advices during these years. Furthermore, I want to thank them for motivating me, for continuing pushing me to do my best and for all the opportunities they have offered me these years. Part of this thesis is the result of the joint work with Prof. M. Hewitt, who supervised me during the year I spent in Chicago. I would like to thank him for teaching me the importance of being concise, concrete and of motivating every choice. Many thanks also to Prof. M. Bertocchi for having meticulously coordinated the PhD school during the first year of this PhD cycle and many thanks to Prof. A. Gnudi for the work she is doing as coordinator. I thank the Professors of the Department of Quantitative Methods at the University of Bergamo and the ones of the OR group at the University of Brescia for the precious courses organized during the first PhD year.

Words could never be enough to express gratitude to my parents, who always remind me the importance of having an education and who pushed me towards wider horizons than the ones I come from. Thanks for all the sacrifices they have done in past years to provide me with a good education and for teaching me what is important in life. I feel blessed for their unconditioned love and for their constant presence. My special thanks to Michele for being by my side since years and for being such an incredible life partner. Thanks for always showing me the positive side of things, for motivating me and for your concreteness. Thanks also for your immense patience with me, for our sacrifices, for letting me free to take decisions. Thanks also to the whole Michele's family for always being there and supportive.

Thanks to my grandmothers for supporting me and to my grandfathers, who would be proud of my achievements. I thank all of them for teaching me the values of honesty, humility, commitment and diligence. Thanks also to my aunt, to my uncles and to the little Alice, whom I wish to be driven by the same passion that drove me during my studies.

I thank my friends of a life, for sharing happy and difficult moments while growing up together and for still being there. Last, many thanks to my colleagues of PhD studies for all the crazy moments of study we shared together.

Table of contents

Introduction **1**

1 A two-stage stochastic model for distribution logistics with transshipment and backordering: stochastic vs deterministic solutions **5**

1.1 Introduction **6**

1.2 Problem Description and Formulation **6**

1.3 Computational complexity **9**

1.4 Computational results **11**

1.4.1 Stochastic solution analysis **12**

1.5 Conclusions **13**

2 Effectiveness of the Rolling horizon approach in solving a multi-stage stochastic distribution logistic problem with transshipment and backordering **15**

2.1 Introduction **16**

2.2 Literature review **17**

2.3 Problem description and formulation **19**

2.4 Two particular cases **23**

2.5 Computational results **26**

2.5.1 Instances description **27**

2.5.2 Solving the multi-stage stochastic programming models **28**

2.5.3 Analysis of the (T+1)-stage stochastic model **30**

2.5.4 The Rolling horizon approach **32**

2.6 Managerial insights **37**

2.7 Conclusions **39**

3 A two-stage stochastic optimization model for the Bike sharing allocation and rebalancing problem **40**

3.1 Introduction **41**

3.2 Literature Review 42
3.2.1 Deterministic bikesharing problems 43
3.2.2 Stochastic bikesharing problems 44
3.2.3 Dynamic rebalancing problem 45
3.3 Problem description 45
3.4 A two-stage stochastic programming formulation 46
3.4.1 Newsvendor-based heuristics 51
3.5 Numerical Results 53
3.5.1 State of art of San Francisco bikesharing system 53
3.5.2 Test setting 54
3.5.3 Analyzing the value of uncertainty and the quality of the expected value solution 59
3.5.4 Newsvendor-based heuristics 61
3.5.5 A comparison with the implemented system 65
3.6 Managerial insights 65
3.7 Conclusions and future works 67

4 Workforce production planning under uncertain learning rates 69
4.1 Introduction 70
4.2 Literature Review 71
4.2.1 Learning and worker assignment 71
4.2.2 Cross-training and practicing 72
4.3 Production setting and managerial tactics 74
4.3.1 Production setting 74
4.3.2 Managerial tactics 76
4.4 Methodology 77
4.4.1 A two-stage stochastic programming model 77
4.4.2 Production planning indicators 80
4.4.3 Linear regression models 81
4.5 Experimental setting 82
4.5.1 Instance parameter values 83
4.5.2 Size of the scenario tree 85
4.5.3 Number of instances 86
4.6 Results and analysis 87
4.6.1 Value in modeling uncertain learning rates 87

4.6.2 Regression results for hypotheses testing . . . 89
4.6.3 Which types of workers should produce more often? . . . 92
4.6.4 Which types of workers should produce more products? . . . 93
4.6.5 Which types of workers should cross-train and which should practice? 93
4.6.6 Tactics for accommodating uncertainty in worker learning rates . . . 94
4.7 Conclusions and future works . . . 95

5 Optimization driven monotonic bounds for two-stage stochastic integer programs 97
5.1 Introduction . . . 98
5.2 Scenario grouping in refinement levels . . . 98
5.2.1 Notation and Preliminaries . . . 98
5.2.2 MIP formulation for scenario grouping . . . 102
5.3 Computational Results . . . 105
5.3.1 Two-stage stochastic optimization integer model for the Bike sharing allocation and rebalancing problem . . . 105
5.3.2 SSLP 10 50 50 . . . 111
5.4 Conclusions . . . 113

Appendices . . . 116
Appendix 1 . . . 116
1.a Model linearization . . . 116
Appendix 2 . . . 117
2.a Model linearization . . . 117
2.b Determining the initial bike requirement for each station . . . 118
2.c The Sequence based Heuristic (SBH) . . . 119
Appendix 3 . . . 122
3.a Reformulation . . . 122
3.b Detailed regression results . . . 124

References . . . 129

List of figures . . . 138

List of tables . . . 141

Introduction

Logistics is the collection of the activities devoted to managing the flows of goods and providing services in an efficient way. Consequently, it involves various and different operations, such as supply, production, stock and distribution activities. Its objectives can be summarized by the so-called "7 R": the Right product (or service), in the Right quantity, at the Right place, at the Right time, at the Right costs, at the Right conditions, for the Right customer. The U.S.A government estimated that, in 2016, the logistics and transportation industry represented 7.5% of U.S.A. GDP (International Trade Administration [47] (2017)). As a matter of fact, nowadays, as the competitive pressure is increasing, a good management of logistic activities is fundamental to limit the impact on costs and to provide a good quality service level. Consequently, this aspect has received increasing attention from the involved companies and service providers, who struggle to effectively deal with strategical, tactical and operational problems ever growing in terms of dimension and complexity.

Traditionally, most of the extant literature has focused on logistics problems by representing and solving them through deterministic optimization models, in which all parameters values are supposed to be known. However, logistics problems are typically characterized by highly dynamic information processes, and, consequently, by uncertainty about the future, as some parameters may be revealed over time. One of the available tools to make decisions that hedge against the future is represented by Stochastic Programming (Birge and Louveaux [12] (2011), King and Wallace [52] (2012), Wallace and Ziemba [90] (2005)) which studies how to incorporate uncertainty into decision-making problems over time. As a matter of fact, there is plenty of real problems in which a decision-maker has to take some decisions immediately, before the realization of a random event, which typically is affecting the outcome of the initial decision. Then, as the uncertainty becomes known period by period, recourse decisions responding to the new information can be made in order to compensate for any undesiderable effects. Concerning the uncertainty, it can be in the models parameters or in the model itself. Specifically, in this work, we consider stochastic parameters which can be due to lack of reliable data, measurement errors, and/or future and unobservable events. Moreover, uncertainty is modeled over time using trees composed of scenarios which approximate and discretize the future realization of the random event. As such, Stochastic Programming turns out to be an effective tool in order to avoid adverse effects caused by uncertainty in the logistic industry, to ensure the regularity of activities and to enhance flexibility to minimize costs.

In this thesis, the first four chapters are devoted to the study of four different problems. The common feature of all these problems is the presence of uncertainty in some parameters. Across all these applications, we show that the solutions of Stochastic Programming models provide efficient policies, because they explicitely model the value of future decisions that are taken after uncertainty has re-

vealed. We also highlight managerial insights which can be valuable for practitioners. Moreover, as stochastic integer programs may be very difficult to solve (even with only two-stages), we also present some preliminary results concerning a methodology to obtain monotonic chains of lower bounds for these programs.

The thesis is organized as follows.

In Chapter 1, a problem arising in distribution logistics with transshipment and backordering is analyzed. With respect to the extant literature, the aim of this chapter is to formulate a new stochastic optimization model for this problem and to provide complexity results. Furthermore, we analyze how stochasticity influences different configurations of a distribution system and we show in which cases the solution of the deterministic problem represents a good starting point in order to solve the stochastic model to optimality, with less time effort.

Chapter 2 presents an extension of the problem presented in Chapter 1 to the multi-stage case. After formulating a multi-stage stochastic optimization model, we present two polynomially solvable cases. As the complexity increases with the number of considered periods, for the two polynomially solvable cases we provide optimal policies, while for the general case, the performance of a rolling horizon approach is tested. We finally report some sensitivity results and managerial insights.

In Chapter 3, an allocation and rebalancing bike sharing problem with stochastic demand is investigated. With respect to the extant literature, we stress the role of uncertainty and we first show the benefits of solving our stochastic program with respect to the solution of the deterministic equivalent formulation. Then, we compare the solution of the stochastic program to the solutions obtained through heuristics based on Newsvendor models. Finally, we benchmark our approach on a real bike-sharing system.

Chapter 4 studies a workforce allocation problem considering that workers learn according to stochastic rates. With respect to the extant literature, we investigate how uncertainty in learning rates impact on the choices of assignments, cross-training and practice. As a matter of fact, nowadays, managers can take decisions about workforce scheduling by exploting data analysis deeply, rather than using the traditional ways based upon personal relationships, personal experience and feelings. This new field is referred to as "People analytics", in which the methods of analytics can help decision-makers in managing employees or workforce. Thanks to the combination of technologies, statistics and mathematical models applied to large sets of data (i.e. "big data"), managers make better decisions in the companies, which are interested in increasing the return on their investment in people.

Finally, Chapter 5 introduces a methodology which allows us to build monotonic chains of lower bounds for two-stage stochastic integer programs which are difficult to solve. Specifically, with respect to the extant literature, we derive a mixed integer model according to which it is possible to build

disjoint scenario groups. The criterion according to which scenarios groups are built is based on the maximization of the estimated improvement obtained by solving the stochastic program on scenarios subgroups with respect to the wait-and-see solution. Some preliminary results for benchmark instances from the literature are presented.

Chapter 1

A two-stage stochastic model for distribution logistics with transshipment and backordering: stochastic vs deterministic solutions

Authors: Rossana Cavagnini[1], Luca Bertazzi[2] and Francesca Maggioni[3]

(Accepted for publication in "New Trends in Emerging Complex Real Life Problems", International Conference on Optimization and Decision Science (ODS) 2018, Airo Springer Series. Manuscript Reference number: ODS2018_015)

Keywords: Optimization under Uncertainty, Transshipment, Backordering, Stochastic solution analysis

[1]University of Bergamo, Via dei Caniana, 2, Bergamo, Italy, e-mail: r.cavagnini@studenti.unibg.it
[2]University of Brescia, Contrada Santa Chiara, 50, Brescia, Italy, e-mail: luca.bertazzi@unibs.it
[3]University of Bergamo, Via dei Caniana, 2, Bergamo, Italy, e-mail: francesca.maggioni@unibg.it

1.1 Introduction

In recent years, competition pressure has increased and logistics has become more and more crucial for the success of companies due to its impact on costs and service levels. An efficient distribution system is fundamental to satisfy customers' requests with reduced lead times and with a good service level. Traditionally, the distribution network is organized as a hierarchical process in which the flow of goods is shipped from the uppermost level of the distribution chain to the lowest. One of the purposes of this paper is to study a more flexible distribution network, where the shipment of products between locations at the same level of the distribution system is admitted. This strategy is called *transshipment* and it allows companies to reduce stock out risks, to share surplus stocks and to improve warehouses management, coping with demand uncertainty.

Based on the inventory system, ordering and transshipment characteristics, Paterson, Kiesmüller, Teunter, and Glazebrook [73] present a complete review of the transshipment literature. Examples of stochastic transshipment problems are Herer and Rashit [40] (1999), where fixed replenishment costs are taken into account, while Olsson [71] (2010) considers the unidirectional transshipment problem, where locations have different backordering and stockout costs. Backordering is not considered in Wee and Dada [91] (2005), while Yücesan et al. [93] (2012) studies the multi-location transshipment problem including lead times. Finally, Rottkemper, Fischer, and Blecken [81] (2012) proposes a stochastic transshipment model for humanitarian emergencies.

Our contribution is to provide insights about the importance of considering uncertainty in a distribution system with transshipment and backordering.

The remainder of the paper is organized as follows. Section 1.2 presents the problem description and formulation. Section 1.4 shows our computational results and, finally, in Section 1.5, conclusions and research perspectives are outlined.

1.2 Problem Description and Formulation

The analyzed problem deals with a *single echelon* distribution system composed of a single supplier and a set $\mathcal{I}$ of M retailers with a *centralized decision making*. Transshipment is admitted and, in order to keep track of the origin and destination of product flows, we represent retailers performing transshipment by index i and retailers receiving transshipped quantities by index j ($i \in \mathcal{I}, j \in \mathcal{I}$). In this problem transshipment is *intra-level* (since it involves only retailers), *bi-directional* (each retailer can both transship products to other retailers and receive products from them) and *reactive* (it is performed in emergency situations, after demand realization). We deal with a *single product complete pooling* transshipment (retailer i can not keep any inventory quantity if retailer j has a shortage of

product), where the *priority principle* is respected (each retailer satisfies its demand at first and then transshipment is performed if necessary), backordering to supplier is allowed and, consequently, the demand can potentially be covered with supplied quantities, with transshipment quantities and with backordered quantities. The unsatisfied demand represents a lost sale. Since retailers are supposed to be close to each other, lead times are considered negligible. Our problem is described on two time intervals: t_0, which represents the time at which we have to take the decision about the quantities to ship from the supplier to retailers and t_1, in which, after demand realization, we decide the quantities to transship and the quantities to backorder.

Moreover, the problem is characterized by risk presence: the demand is a phenomenon which can not be exactly forecast, but it is stochastic. We denote by d all possible values for the demand, that is a random variable having discrete (mutually indipendent) probability distributions $\mathcal{D}_i$, defined over the support $\mathcal{U}_1 = \{\underline{d}, \ldots, \overline{d}\}$, where $0 < \underline{d} \leq \overline{d}$. Furthermore, we represent by $\mathcal{S}$ the set of scenarios s, $s = 1, \ldots, S$ and by pr^s the probability of each scenario $s \in \mathcal{S}$, so that d_i^s denotes the demand realization for retailer i in scenario s. The measure adopted to evaluate the system performance is the total expected cost.

At time t_0, the decision variables of this model are x_i, which represent the decisions to take at the first stage, i.e. the quantity to ship from the supplier to each retailer i, taking into account the supplier's total inventory availability q and the associated unit inventory cost h_0. We introduce a capacity C_i for each vehicle employed in the shipment of units from the supplier to retailer i and an integer variable v_i, standing for the number of total vehicles used to serve retailer i by direct shipping. The transportation cost between the supplier and each retailer is represented by a variable cost f_i, proportional to the number of shipped units and by a fixed component F_i, paid for each vehicle used.

If retailer j has to face a demand d_j^s greater than the initial inventory level $\overline{I}_{i0}$ plus the quantity x_i received from the supplier, transshipment and/or backordering can be used to avoid stock-out. Thus, at t_1 the decision variables are represented by y_{ij}^s which stand for the quantity to transship from retailer i to retailer j, for each possible scenario s, after the demand realization d_i^s and by b_i^s which represent the quantity to backorder from the supplier for each retailer and for each possible scenario s, after demand realization d_i^s. On one hand, we introduce a capacity C^T for vehicles used to transship units (note that the capacity of vehicles used to ship units from supplier to retailers is typically bigger than the capacity of vehicles used for transshipment) and integer variables V_{ij}^s representing the number of vehicles employed for transshipment from retailer i to retailer j for each scenario s. The total transshipment cost is composed of a unit cost t_{ij} for each transshipped unit and a fixed cost T_{ij} for each vehicle used. On the other hand, backordering is done by using vehicles with the same capacity C_i of vehicles used for the shipment from the supplier to retailer i and we represent the number of

vehicles used for backordering with the variables r_i^s. The total backordering cost is composed of a unit backordering cost g_i for each backordered unit and a fixed cost G_i for each vehicle used. Finally, the variables I_i^s represent the balance quantity at each retailer i for each scenario s and they are given by the sum of the initial inventory level $\overline{I}_{i0}$ plus the quantity received from the supplier, the quantity received through transshipment and through backordering minus the sum of the customers' demand and of the transshipped units. If this quantity is positive, it stands for the inventory level and the associated unit cost is represented by h_i. If the quantity is negative, then the balance quantity stands for the stock-out quantity and retailer j has to pay a unit penality cost p_j. In particular, if the product surplus at retailer i is transshipped to retailer j, but it is not sufficient to fully cover the shortage of product of retailer j, and no quantities are backordered, retailer i has neither inventory nor stock-out costs, while retailer j has to face stock-out costs for the unsatisfied demand. We also consider the warehouse capacity Q_i for each retailer i.

Consequently, we formulate the following integer non linear two stage stochastic programming model.

Model $\mathcal{T}$

$$\begin{aligned}\min\ & h_0(q-\sum_{i\in\mathcal{I}}x_i)+\sum_{i\in\mathcal{I}}(f_ix_i+F_iv_i)+\\ &+\sum_{s\in\mathcal{S}}pr^s[h_0(q-\sum_{i\in\mathcal{I}}x_i-\sum_{i\in\mathcal{I}}b_i^s)+\sum_{i\in\mathcal{I}}(g_ib_i^s+G_ir_i^s)+\\ &+\sum_{i\in\mathcal{I}}h_i\max\{I_i^s,0\}+\sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{I}:i\neq j}(t_{ij}y_{ij}^s+T_{ij}V_{ij}^s)-\sum_{j\in\mathcal{I}}p_j\min\{I_j^s,0\}]\end{aligned} \tag{1.1}$$

s.t.

$$\sum_{i\in\mathcal{I}}(x_i+b_i^s)\leq q \qquad s\in\mathcal{S} \tag{1.2}$$

$$I_i^s=\overline{I}_{i0}+x_i+b_i^s-d_i^s+\sum_{j\in\mathcal{I}:i\neq j}(y_{ji}^s-y_{ij}^s) \qquad i\in\mathcal{I},\ s\in\mathcal{S} \tag{1.3}$$

$$I_i^s\leq Q_i \qquad i\in\mathcal{I},\ s\in\mathcal{S} \tag{1.4}$$

$$x_i\leq C_iv_i \qquad i\in\mathcal{I} \tag{1.5}$$

$$b_i^s\leq C_ir_i^s \qquad i\in\mathcal{I},\ s\in\mathcal{S} \tag{1.6}$$

$$y_{ij}^s\leq C^TV_{ij}^s \qquad i\in\mathcal{I},\ j\in\mathcal{I}:j\neq i,\ s\in\mathcal{S} \tag{1.7}$$

$$x_i\geq 0\ integer \qquad i\in\mathcal{I} \tag{1.8}$$

$$y_{ij}^s\geq 0\ integer \qquad i\in\mathcal{I},\ j\in\mathcal{I}:j\neq i,\ s\in\mathcal{S} \tag{1.9}$$

$$b_i^s\geq 0\ integer \qquad i\in\mathcal{I},\ s\in\mathcal{S} \tag{1.10}$$

$$v_i\geq 0\ integer \qquad i\in\mathcal{I} \tag{1.11}$$

$$r_i^s \geq 0 \; integer \qquad i \in \mathcal{I},\ s \in \mathcal{S} \tag{1.12}$$

$$V_{ij}^s \geq 0 \; integer \qquad i \in \mathcal{I},\ j \in \mathcal{I} : j \neq i,\ s \in \mathcal{S} \tag{1.13}$$

$$I_i^s \; free \qquad i \in \mathcal{I},\ s \in \mathcal{S} \tag{1.14}$$

where the objective function (1.1) represents the minimization of the total expected cost, obtained through the sum of the supplier's inventory cost, the total shipment costs from supplier to retailers, the expected supplier's inventory costs, the total expected backordering cost, the total expected retailers' inventory cost, the total expected transshipment costs and the expected stock-out costs. Constraints (1.2) implies that the total quantity shipped from the supplier to all retailers (through usual shipment and backordering) cannot be greater than the supplier's initial inventory. Constraints (1.3) are the balance constraints. Constraints (1.4) imply that the balance quantity (computed as in (1.3)) cannot exceed the warehouse capacity Q_i for each retailer i. Constraints (1.5), (1.6) and (1.7) link together the decision variables x_i, b_i^s and y_{ij}^s with the respective integer variables v_i, r_i^s and V_{ij}^s so that if the first ones are positive, these quantities are splitted in a certain number of vehicles represented by the latter ones, considering the respective vehicles capacities C_i and C^T and, consequently, the associated fixed costs F_i, G_i and T_{ij} are charged in the objective function. Finally, constraints from (1.8) to (1.14) are variables definition constraints. Due to the non-linearity of Model $\mathcal{T}$, we linearize it following the approach described in Cavagnini, Bertazzi, Maggioni, and Hewitt [19] (2018) and we call the linearized problem "Model $\mathcal{T}^{\mathcal{L}}$".

1.3 Computational complexity

In this section, we prove the computational complexity of Model $\mathcal{T}^{\mathcal{L}}$.

Theorem 1. *Model $\mathcal{T}^{\mathcal{L}}$ is NP-hard.*

Proof. Consider the set of istances such that the demand for each retailer is deterministic and represented by $\overline{d}_i$, with $\sum_{i\in\mathcal{I}} \overline{d}_i = q$, $\bar{I}_{i0} = 0$, $C_i \geq \max_{i\in\mathcal{I}}\{\overline{d_i}\}$, $C^T \geq \sum_{i\in\mathcal{I}} \overline{d_i}$ and $Q_i = \infty$. Moreover, suppose that the supplier's inventory cost $h_0 = 0$, the inventory and stock-out costs $h_i = \infty$ and $p_j = \infty$, respectively, and that the backordering and transshipment fixed and variable costs $G_i = \infty$, $g_i = \infty$, $T_{ij} = \infty$, $t_{ij} = \infty$, respectively. Then, Model $\mathcal{T}^{\mathcal{L}}$ becomes:

$$\begin{aligned} \min \ & \sum_{i\in\mathcal{I}} (f_i x_i + F_i v_i) + \sum_{i\in\mathcal{I}} (g_i b_i + G_i r_i) + \\ & + \sum_{i\in\mathcal{I}} h_i \max\{I_i, 0\} + \sum_{i\in\mathcal{I}} \sum_{j\in\mathcal{I}:i\neq j} (t_{ij} y_{ij} + T_{ij} V_{ij}) - \sum_{j\in\mathcal{I}} p_j \min\{I_j, 0\}] \end{aligned} \tag{1.15}$$

s.t.

$$\sum_{i\in\mathcal{I}}(x_i + b_i) \leq q \tag{1.16}$$

$$I_i = \overline{I}_{i0} + x_i + b_i - \overline{d}_i + \sum_{j\in\mathcal{I}:i\neq j}(y_{ji} - y_{ij}) \qquad i \in \mathcal{I} \tag{1.17}$$

$$I_i^s \leq Q_i \qquad i \in \mathcal{I},\ s \in \mathcal{S} \tag{1.18}$$

$$x_i \leq C_i v_i \qquad i \in \mathcal{I} \tag{1.19}$$

$$b_i \leq C_i r_i \qquad i \in \mathcal{I} \tag{1.20}$$

$$y_{ij} \leq C^T V_{ij} \qquad i \in \mathcal{I},\ j \in \mathcal{I} : j \neq i \tag{1.21}$$

$$x_i \geq 0\ integer \qquad i \in \mathcal{I} \tag{1.22}$$

$$y_{ij} \geq 0\ integer \qquad i \in \mathcal{I},\ j \in \mathcal{I} : j \neq i \tag{1.23}$$

$$b_i \geq 0\ integer \qquad i \in \mathcal{I} \tag{1.24}$$

$$v_i \in \{0,1\} \qquad i \in \mathcal{I} \tag{1.25}$$

$$r_i \in \{0,1\} \qquad i \in \mathcal{I} \tag{1.26}$$

$$V_{ij} \in \{0,1\} \qquad i \in \mathcal{I},\ j \in \mathcal{I} : j \neq i \tag{1.27}$$

$$I_i\ free \qquad i \in \mathcal{I} \tag{1.28}$$

which is equivalent to the following *Fixed Charge Transportation Problem* (see Klose [53] (2006) and Roberti, Bartolini, and Mingozzi [79]) (2014):

$$\min \sum_{i\in\mathcal{I}}(f_i x_i + F_i v_i) \tag{1.29}$$

$$x_i = \overline{d}_i \qquad i \in \mathcal{I} \tag{1.30}$$

$$\sum_{i\in\mathcal{I}} x_i = \sum_{i\in\mathcal{I}} \overline{d}_i = q \tag{1.31}$$

$$0 \leq x_i \leq C_i v_i \qquad i \in \mathcal{I} \tag{1.32}$$

$$v_i \in \{0,1\} \tag{1.33}$$

which is known to be NP-hard. □

1.4 Computational results

Model $\mathcal{T}^{\mathcal{L}}$ was implemented in Python 3.6.1 using the Gurobi 7.5.1 solver, and run on an Intel Core i7-7500U 2.70 GHz and 8GB RAM personal computer. Due to the complexity of Model $\mathcal{T}^{\mathcal{L}}$, the running is stopped when a 1% relative gap to the optimal solution or a time limit of 1 hour is reached. We first consider the case with two retailers (i.e. $\mid \mathcal{I} \mid$=2). Our instances are inspired by a real case presented in Bertazzi and Maggioni [11] (2018), in which the uncertain demand of pallets should be satisfied by using trucks with limited capacity. The support of the demand probability distribution is in the set of integer numbers in the interval $[30, 130]$, while the probability distribution is given by a Beta distribution (α, β), where α=20 and β=16, having average demand $\mathbb{E}(d) = 85.55556$ pallets. The supplier's inventory level q is equal to 200 pallets, the capacity C_i of the vehicles used for shipment and backordering to all retailers is equal to 34 pallets, the capacity C^T of the vehicle used for transshipment is 17 pallets, while the retailers' warehouse capacity Q_i is equal to 170 pallets. Furthermore, we define the value P of a pallet to be equal to 1053 Euros, and since the unit inventory costs approximatively correspond to 5% of the value of a pallet of 100 kilograms, we set the supplier's inventory cost equal to 5% P, and the retailers' inventory costs equal to 6% P. Moreover, since the penalty cost corresponds to a lost sale and to a reputation damage, we let p_j equal to 1.5 P. As in Bertazzi and Maggioni [11] (2018), we consider a unit shipment cost of a pallet with 100-200 kilograms weight on a distance up to 500 kilometers equal to 93.60 Euros and a fixed shipment cost equal to $\frac{f_i C_i}{\theta}$, where $\theta = 0.5$. Finally, considering that the fixed transshipment and backordering costs are computed as a function of the unit transshipment and backordering costs, 25 different instances are generated by combining all possible values, as displayed in Table 1.1. We notice that Model $\mathcal{T}^{\mathcal{L}}$ can be reduced into different special cases, which facilitate a trade-off analysis. In particular, in the "Extremely High case", obtained by assigning to transshipment and backordering costs a very high value (for example, equal to infinity), we get one instance in which both transshipment and backordering are not allowed, four instances in which only backordering is allowed and four instances in which only transshipment is allowed. The same parameters are considered also in the case with four retailers, (i.e. $\mid \mathcal{I} \mid = 4$), apart from q which is equal to 350 pallet.

Cost	Extremely Low case (EL)	Low (L)	Medium (M)	High (H)	Extremely High case (EH)
t_{ij}	0	$\frac{0.75 f_i}{2} = 35.1$	$\frac{f_i}{2} = 46.8$	$\frac{1.25 f_i}{2} = 58.5$	$+\infty$
T_{ij}	0	$\frac{t_{ij} C^T}{0.5} = 1193.4$	$\frac{t_{ij} C^T}{0.5} = 1591.2$	$\frac{t_{ij} C^T}{0.5} = 1989$	$+\infty$
g_i	0	$0.75 f_i = 70.2$	$f_i = 93.6$	$1.25 f_i = 117$	$+\infty$
G_i	0	$\frac{g_i C}{0.5} = 4773.6$	$F_i = 6364.8$	$\frac{g_i C}{0.5} = 7956$	$+\infty$

Table 1.1: Transshipment and backordering fixed and unit costs

In order to determine the right number of scenarios which have to be considered for the stochastic setting, we perform the in-sample stability analysis identifying as benchmark scenario tree, the one with 500 scenarios. The out-of-sample stability analysis in the benchmark tree is obtained with 300 scenarios.

1.4.1 Stochastic solution analysis

In this section, we perform the stochastic solution analysis considering the benchmark scenario tree with 500 scenarios and computing the indicators presented in Maggioni and Wallace [56] (2012). Table 1.2 displays the average results for the two retailers case, where with "Other" we refer to instances not belonging to any special case (i.e. the ones in which both transshipment and backordering are allowed). First, the availability of a perfect information about the future is more important if recourse decisions (i.e. backordering and transshipment) are not allowed or just transshipment is admitted with an $EVPI$ of 12.07% in the first case and approx. 10% in the second. The case in which only backordering is allowed is the most flexible with an $EVPI$ of 1.72%, as new quantities can be introduced in the system through the recourse decision, while when only transshipment is allowed, there can be a flow of goods between retailers, but further quantities are not available. Concerning the Value of Stochastic Solution, *VSS*, results show there are more advantages in including stochasticity in the cases where no recourse actions are admitted or only less flexible recourse actions are allowed (i.e. transshipment). In order to understand why the deterministic solution is worse compared to the stochastic one, we compute the $LUSS$ and the $LUDS$ indicators. Through the $LUSS$, we see that in the cases where no recourse decisions or just one of them are admitted, the deterministic solution identifies the same retailers selected by the stochastic solution, but with wrong delivered quantities. In the other cases, the retailers receiving zero quantities are different in the stochastic and in the deterministic solution and, as a consequence, the poor performance is due both to the selection of retailers and to the selection of the quantities. Through the $LUDS$, we notice that the solution is perfectly upgreadable only if both backordering and transshipment are not allowed, meaning that these quantities are always lower or equal to the ones suggested by the stochastic program. For all other cases, the $LUDS$ is not null, meaning that the deterministic solution is only partially upgreadable (at least in one case, the stochastic solution delivers a lower number of pallets than the one suggested by the deterministic solution).

Finally, we focus on the case with four retailers. Due to the computational complexity of the problem, with the exception of the case "No transshipment, No backordering", we analyze only the instances whose costs of the allowed strategy are set at a "Medium" level (i.e. only one instance for each case is considered). Results are displayed in Table 1.3. We specify that after 549090 seconds, the gap to

Cases	RP	WS	EVPI	EEV	VSS	ESSV	LUSS	EIV	LUDS
No transshipment No backordering	56941.68	50066.85	12.07%	57688.54	1.31%	56941.68	0.00%	56941.68	0.00 %
Only backordering	40856.24	40153.84	1.72%	40956.26	0.25%	40856.24	0.00%	40876.56	0.05%
Only transshipment	53557.80	48337.35	9.75%	54600.35	1.95%	53557.80	0.00%	53567.35	0.02%
Other	39487.43	38979.73	1.29%	39723.29	0.60%	39512.12	0.06%	39504.06	0.04%

Table 1.2: Average values for the stochastic solution analysis indicators for every special case with two retailers

the optimal solution of the *RP* for the "Other" case was not closed and we calculate only the *EVPI* and the *VSS*, since the other indicators require further constraints which make the model even more difficult to get solved to optimality. Differently from Table 1.2, now, if only backordering is allowed the cost is higher than the case in which only transshipment is admitted, while for the *EVPI*, the previous results are confirmed. Concerning the VSS, the results are now different, as there are more advantages in including stochasticity in the case where only backordering is allowed. Even if with only backordering, the quantities delivered in the first stage are fewer, transshipment is cheaper if only few quantity adjustments are needed and the presence of more retailers provides more flexibility to the distribution system.

Cases	RP	WS	EVPI	EEV	VSS
No transshipment No backordering	120144.82	109191.65	10.03%	131098.04	9.12%
Only backordering	112319.31	109191.65	2.86%	122822.55	9.35%
Only transshipment	107613.10	103077.98	4.40%	111960.39	4.04%
Other	105432.62 (2.27%)	103077.98	2.28%	106535.91	1.05%

Table 1.3: Values for the stochastic solution analysis indicators for every special case with four retailers and "Medium" cost level

1.5 Conclusions

We presented a real problem arising in logistics and after modeling it with an integer stochastic program, we stated that this is NP-hard. Furthermore, we show that with two retailers, a decision-maker has a greater advantage by including uncertainty, especially if no recourse actions or only transshipment is admitted. We also show that in some cases, the selection of retailers to which quantities should be delivered is the same both in the deterministic and in the stochastic solution.

Nevertheless, the deterministic solution can be upgrated only in the special case where no recourse actions are allowed. Conversely, with four retailers, transshipment provides more flexibility. Future research could be devoted to analyze the multistage version of this problem by exploiting lower bounds (see Maggioni, Allevi, and Bertocchi [58] (2016), Maggioni and Pflug [55] (2016)) and, as in Bertazzi and Maggioni [11] (2018), to compare the stochastic solution to the one obtained through a rolling-horizon heuristic. Another stream of research could be analyzing robust optimization approaches (see Maggioni, Potra, and Bertocchi [59] (2017)) or adapting approaches presented in Bertazzi and Maggioni [10] (2015).

Chapter 2

Effectiveness of the Rolling horizon approach in solving a multi-stage stochastic distribution logistic problem with transshipment and backordering

Authors: Rossana Cavagnini[1], Francesca Maggioni[2], Luca Bertazzi[3]

Keywords: Logistics, Fixed-charge transshipment problem, Multi-stage stochastic programming, Rolling horizon approach

[1]University of Bergamo, Via dei Caniana, 2, Bergamo, Italy, e-mail: r.cavagnini@studenti.unibg.it
[2]University of Bergamo, Via dei Caniana, 2, Bergamo, Italy, e-mail: francesca.maggioni@unibg.it
[3]University of Brescia, Contrada Santa Chiara, 50, Brescia, Italy, e-mail: luca.bertazzi@unibs.it

2.1 Introduction

In recent years, competition pressure has increased and an effective management of distribution logistics has become fundamental for companies' profitability, due to its impact on costs and service levels. As a consequence, the efficient distribution system should satisfy customers' requests, while keeping costs as low as possible. Traditionally, distribution networks are organized as a hierarchical process, in which the products are shipped from the uppermost level of the distribution chain to the lowest. Nevertheless, physical pooling of inventories (Eppen [30] (1979)) is frequently used by practitioners to achieve the best trade-off between customers' satisfaction and cost reduction. This strategy has also been referred to as "transshipment"; it allows companies to reduce stock-out risks, to share surplus stocks and to improve warehouses management, coping with demand uncertainty.

In this paper, we study a distribution system with uncertain demand, where each retailer has to decide the quantities to receive through usual shipment and, then, after the realization of the demand, how many units to receive through transshipment and backordering.

This problem arises in different contexts in the real-world. For example, companies whose distribution operations are in charge of retailers are interested in satisfying as much demand as possible to increase profits and to keep the company reputation high by offering a good service level. Furthermore, they desire to keep the inventory levels as low as possible, because of the associated costs and because of the obsolescence risk (especially for fashion and technology industries). These two objectives are even more important for e-commerce companies (for example, Amazon) in which there is a big central warehouse and some regional depots and the demand for fast deliveries is increasing. In order to prevent stockouts, backordering is a useful strategy, but it has two main drawbacks. The first is that it does not contribute to rebalance inventory levels through retailers. The second is that, the supplier or the central warehouse is often located far from the retailers or from the regional depots. Consequently, transshipment is a valid strategy to overcome these pitfalls. On one hand, through transshipment, retailers can share inventory quantities. On the other hand, transshipment allows retailers to limit stockouts through quicker shipments than the ones originating from the supplier or from the central warehouse.

Motivated by the stochastic and dynamic nature of this problem, we formulate a multi-stage stochastic program. However, it is well known that solving a problem over a long time horizon can be very time consuming. As such, heuristic methods, such as the Rolling horizon, in which the time horizon is decomposed into periods of shorter length, give rise to subproblems easier to solve. In this paper, we test the performance of a Rolling horizon approach. In a rolling horizon decision-making framework, a manager must decide the length of the reduced time horizon. In making this choice, his/her main objectives are (1) the reduction of the computational time (as he/she would like to be able to solve

the problem as quickly as possible) and (2) the minimization of the increase in the objective function value due to a suboptimal first-stage decision (a phenomenon which is also known as "*end-of-horizon effect*"). These two objectives become even more important if decisions must be taken day-by-day (at a tactical/operational level). Furthermore, understanding the length of the reduced time horizon which should be taken into account is valuable, as further evaluations become unnecessary (Chand, Hsu, and Sethi [20] (2002)).

The main contributions of this paper with respect to the extant literature are the following. First, a new problem (a multi-stage stochastic fixed charge distribution problem with transshipment and backordering) is studied, and a new multi-stage stochastic optimization model for this problem is formulated. Second, two polynomial solvable cases are derived. Third, through a systematic and extensive computational study, the maximum dimension of the multi-stage stochastic programming models which can be solved by an off-the-shelf solver (both for the polynomial and for the general cases) is provided and the sensitivity of the optimal policies to the number of periods, to the unit inventory costs and to the vehicles capacities is studied. Finally, the performance of the Rolling horizon approach is benchmarked against optimal solutions, also by considering different inventory unit costs and different reduced time horizons.

The paper is organized as follows. In Section 2.2, the extant literature with respect to the main contributions of our work is reviewed. Section 2.3 presents the problem description, as well as the model formulation. In Section 2.4, we provide two polynomially solvable cases, namely the one in which only backordering is allowed, and the one in which the warehouses are always undersized with respect to the demand. Section 2.5 shows the computational results, Section 2.6 provides some managerial insights. Finally, in Section 2.7, we draw some conclusions and describe the future developments of our work.

2.2 Literature review

Multiple different transshipment problems and variants have been studied in the literature. Paterson, Kiesmüller, Teunter, and Glazebrook [73] (2011) present a complete review of the transshipment literature, based on the inventory system, ordering and transshipment characteristics. According to their classification, our problem studies a single echelon distribution system with centralized decision-making and composed of two different locations dealing with a single commodity. Moreover, in their categorization, transshipment is classified as reactive, complete pooling and the associated cost structure includes both variable and fixed costs. Backordering and lost sales are allowed too. Concerning the characteristics of the problem we study, one of the main distinguishing features lies in the consideration

of both fixed and variable transshipment costs. Other papers have previously considered these features (see, for example, Robinson [80] (1990), Chang and Lin [21] (1991), Herer and Rashit [40] (1999), Herer and Tzur [41] (2001), Herer and Tzur [42] (2003), Chiu and Huang [23] (2003), Minner, Silver, and Robb [67] (2003), Axsäter [6] (2003), Minner and Silver [66] (2005)). Nevertheless, all these papers differ from ours as some consider a decentralized decision-making, and/or partial pooling, and/or no backordering, and/or identical depots. Furthermore, differently from the literature, in which typically it is assumed that backordering and transshipment are outsourced or performed by owned vehicles, we compare both the two options.

A stream of research has also addressed the stochastic transshipment problems. Rottkemper, Fischer, and Blecken [81] (2012) consider a distribution and inventory relocation problem in humanitarian operations with uncertainty in demand. Specifically, they focus on the minimization of the unsatisfied demand and perform a sensitivity analysis by using a rolling horizon solution method, motivated by the dynamic nature of the analyzed problem. Moreover, Yücesan et al. [93] (2012) study a problem with non-negligle replenishment lead times in a distribution system with stochastic demand and derive a duality-based gradient method to improve computational efficiency for examining the multi-location transshipment problem. Mirzapour Al-e hashem, Rekik, and Hoseinhajlou [68] (2017) study the environmental impacts related to an Inventory routing problem with transshipment and stochastic demand. Finally, Cavagnini, Bertazzi, and Maggioni [18] (2018) present a two-stage stochastic program for a distribution system with transshipment and backordering and provide a comparison between the stochastic and the determinstic solution, in the case with two and four retailers. They show that, by explicitely considering uncertainty in demand, a decision-maker can make better decisions.

Another group of research has focused on deriving optimal ordering and/or replenishment policies where transshipment is allowed. Among these papers, we mention Wee and Dada [91] (2005), Özdemir, Yücesan, and Herer [72] (2006) (in which vehicles capacities are considered too), Herer, Tzur, and Yücesan [43] (2006), Feng, Moon, and Ryu [34] (2017) and van Wijk, Adan, and van Houtum [89] (2018).

The computational complexity of transshipment problems has seldom been studied in the literature. To the best of our knowledge, only Herer and Tzur [42] (2003) provide a formal proof of the NP-hardness of a deterministic model with transshipment, but without backordering. Moreover, Olsson [71] (2010) and Axsäter [5] (2003) study a unidirectional lateral transshipment problem, motivated by the advantage of being much less complex than a system in which transshipment is allowed between all locations. However, we believe that this unidirectional flow structure rarely fits to real-world problems. Consequently, the problem should be considered in all its complexity, and heuristic algorithms become needed and valuable. In this work, we test the performance of a rolling horizon approach. This

approach has been extensively used in the literature (see Chand, Hsu, and Sethi [20] (2002) for a classified bibliography of the literature). Some examples of its application to transportation problems are Maggioni, Kaut, and Bertazzi [57] (2009), Shen, Chu, and Chen [84] (2011) and Bertazzi and Maggioni [11] (2018), which also provide a worst-case analysis of the rolling horizon approach.

2.3 Problem description and formulation

The analyzed problem deals with a *single echelon* distribution system composed of a single supplier and a set $\mathcal{I}$ of M retailers with a *centralized decision making*. Transshipment between pairs of retailers is admitted and, in order to keep track of the origin and destination of product flows, we represent retailers performing transshipment by index i and retailers receiving transshipped quantities by index j ($i \in \mathcal{I}, j \in \mathcal{I}$). In this problem, transshipment is *intra-level* (since it involves only retailers), *bi-directional* (each retailer can both transship products to other retailers and receive products from them) and *reactive* (it is performed in emergency situations, after demand realization). We deal with a *single product complete pooling* transshipment (retailer i cannot keep any inventory quantity if retailer j has a shortage of product), where the *priority principle* is respected (each retailer satisfies its demand at first and then transshipment is performed, if necessary), backordering to supplier is allowed. Consequently, the demand can be satisfied with supplied quantities, with transshipment quantities and with backordered quantities. The unsatisfied demand represents a lost sale. Since retailers are supposed to be close to each other, lead times are considered negligible (not greater than the time unit).

The time horizon is composed of $t \in \mathcal{T} = \{0, \dots, T\}$ discrete time periods. Shipments between supplier and retailers can be done at any of the discrete time periods $t \in \mathcal{T}' = \{0, \dots, T-1\}$, while transshipment and backordering can be performed at $t \in \mathcal{T}'' = \{1, \dots, T\}$ as they are recourse actions. The demand d_i of retailer i is a random variable having discrete (mutually indipendent) probability distribution $\mathcal{D}_i$, defined over the support $\mathcal{U}_1 = \{\underline{d}, \dots, \overline{d}\}$, where $0 < \underline{d} \leq \overline{d}$. The information structure can be described by a scenario tree where, at each stage $t \in \mathcal{T}$, there is a discrete number of nodes $\mid n^t \mid$, where a specific realization of the uncertain demand takes place. There are T levels (stages) in the tree, corresponding to specific time periods. The final $\mid n^T \mid$ nodes are called the leaves. We introduce the set $\mathcal{N}^t$ to describe the set of ordered nodes of the tree at stage $t = 0, \dots, T$. Note that $\mathcal{N}^0$ is composed of a unique node, i.e. the root. Each node at stage t, except the root, is connected to a unique node at stage $t-1$, which is called ancestor node $a(n)$ and to nodes at stage $t+1$, called successors. We denote with $\pi_{a(n),n}$ the conditional probability of the random process at node n given its history up to the ancestor node $a(n)$. A scenario is a path through nodes from the root

node to the leaf node. We represent with π^n the probability of node n (at stage t): if node n at stage t is reachable through node n^1 at stage 1, node n^2 at stage 2, node n^{t-1} at stage $t-1$, then $\pi^n := \pi_{n^1,n^2} \cdot \pi_{n^2,n^3}, \cdot \ldots \cdot \pi_{n^{t-1},n^t}$. Furthermore, $\sum_{n \in \mathcal{N}^t} p^n = 1$. We represent the demand realization for retailer i at node $n \in \mathcal{N}^t$, $t \in \mathcal{T}''$ by d_i^n. We also assume that the probability distributions $\mathcal{D}_i^t$ are stage indipendent and mutually indipendent and that the realization of the random variables in each time period becomes known at the end of the time period.

The measure adopted to evaluate the system performance is the total expected cost. At time periods $t \in \mathcal{T}'$, the decision variables of this model are x_i^n, $n \in \mathcal{N}^t$, which represent the decisions to take at the beginning of each time period, i.e. the quantity to ship from the supplier to each retailer i. We introduce a capacity C for each vehicle employed in the shipment from the supplier to retailer i and an integer variable v_i^n, standing for the number of vehicles used to serve retailer i by direct shipping. The transportation cost between the supplier and each retailer i is represented by a variable cost f_i, proportional to the number of shipped units and by a fixed cost F_i, paid for each vehicle used. If retailer j has to face a demand d_j^n greater than the sum of its initial inventory level and the quantity received from the supplier, transshipment or backordering can be used to avoid stock-out. Thus, at $t \in \mathcal{T}''$ the decision variables are represented by y_{ij}^n, $n \in \mathcal{N}^t$, which stand for the quantity to transship from retailer i to retailer j, for each node at each period $t \in \mathcal{T}''$, after the demand realization d_i^n, and by b_i^n, which represent the quantity to backorder from the supplier for each retailer i for each node $n \in \mathcal{N}^t$ at each each period $t \in \mathcal{T}''$, after demand realization d_i^n. Even though lead times are not considered, we suppose that the time in which the recourse actions are performed is limited and thus, the backordered quantities cannot be used for transshipment in the same time period. For the same reason, we also consider that transshipment is performed between pairs of retailers. We introduce a capacity C^{TB} for vehicles used to transship and to backorder units (note that the capacity of vehicles used to ship units from supplier to retailers is typically larger than the capacity of vehicles used for transshipment and backordering). We define integer variables V_{ij}^n representing the number of vehicles employed for transshipment from retailer i to retailer j for each node $n \in \mathcal{N}^t$ at each stage $t \in \mathcal{T}''$. The total transshipment cost is composed of a unit cost t_{ij} for each transshipped unit and a fixed cost T_{ij} for each vehicle used. We indicate the number of vehicles used for backordering with the variables r_i^n. The total backordering cost is composed of a unit backordering cost g_i for each backordered unit and a fixed cost G_i for each vehicle used. Finally, the variables I_i^n represent the inventory level at each retailer i for each node $n \in \mathcal{N}^t$ at each stage $t \in \mathcal{T}''$ and they are given by the sum of the initial inventory level, plus the quantity received from the supplier, the quantity received through transshipment and through backordering, minus the sum of the customers' demand and of the transshipped units to other retailers. If this quantity is positive, it stands for the inventory level and the associated unit cost is

represented by h_i. If the quantity is negative, then the inventory level stands for a stock-out quantity. In this case, the retailer i has to pay a unit penality cost p_i. In particular, if the product surplus at retailer i is transshipped to retailer j, but it is not sufficient to fully cover the shortage at retailer j, and no quantities are backordered, retailer i has neither inventory nor stock-out costs, while retailer j has to face stock-out costs for the unsatisfied demand. We also consider the warehouse capacity Q_i for each retailer i.

We assume the following notation.

Sets:

$\mathcal{I}$: set of retailers;

$\mathcal{T} = \{t : 0, \ldots, T\}$: set of stages;

$\mathcal{T}' = \{t : t = 0, \ldots, T-1\}$: set of stages (leaf nodes excluded);

$\mathcal{T}'' = \{t : t = 1, \ldots, T\}$: set of stages (root node excluded);

$\mathcal{N}^0 = \{n : n = 0\}$: root node at stage 0;

$\mathcal{N}^t = \{n : n = 1, \ldots, n^t\}$: set of ordered nodes of the tree at stage $t \in \mathcal{T}''$, where n^t is the number of nodes at stage t.

Deterministic parameters:

f_i: unit shipment cost for retailer $i \in \mathcal{I}$;

F_i: fixed shipment cost for retailer $i \in \mathcal{I}$;

g_i: unit backordering cost for retailer $i \in \mathcal{I}$;

G_i: fixed backordering cost for retailer $i \in \mathcal{I}$;

h_i: unit inventory cost for retailer $i \in \mathcal{I}$;

t_{ij}: unit transshipment cost from retailer $i \in \mathcal{I}$ to retailer $j \in \mathcal{I}$;

T_{ij}: fixed transshipment cost from retailer $i \in \mathcal{I}$ to retailer $j \in \mathcal{I}$;

p_j: unit stock-out penalty cost for retailer $i \in \mathcal{I}$;

I_i^0: initial inventory at retailer $i \in \mathcal{I}$, at time $t = 0$;

Q_i: warehouse capacity of retailer $i \in \mathcal{I}$;

C: vehicle capacity for shipment;

C^{TB}: vehicle capacity for transshipment and backordering;

$a(n)$: ancestor of the node $n \in \mathcal{N}^t$, $t \in \mathcal{T}''$ in the scenario tree.

Stochastic parameters:

p^n: probability of node $n \in \mathcal{N}^t$, $t \in \mathcal{T}$;

d_i^n: demand at retailer $i \in \mathcal{I}$ at node $n \in \mathcal{N}^t$, $t \in \mathcal{T}''$.

Variables:

$x_i^n \in \mathbb{Z}^+$: quantity to ship from the supplier to retailer $i \in \mathcal{I}$, for $n \in \mathcal{N}^t$, $t \in \mathcal{T}'$;

$v_i^n \in \mathbb{Z}^+$: number of vehicles used for shipment from the supplier to retailer $i \in \mathcal{I}$, for $n \in \mathcal{N}^t$, $t \in \mathcal{T}'$;

$b_i^n \in \mathbb{Z}^+$: quantity to backorder from the supplier to retailer $i \in \mathcal{I}$, for $n \in \mathcal{N}^t$, $t \in \mathcal{T}''$;

$r_i^n \in \mathbb{Z}^+$: number of vehicles used for backordering from the supplier to retailer $i \in \mathcal{I}$, for $n \in \mathcal{N}^t$, $t \in \mathcal{T}''$;

$y_{ij}^n \in \mathbb{Z}^+$: quantity to transship from retailer $i \in \mathcal{I}$ to retailer $j \in \mathcal{I}$, for $n \in \mathcal{N}^t$, $t \in \mathcal{T}''$;

$V_i^n \in \mathbb{Z}^+$: number of vehicles used for transshipment from retailer $i \in \mathcal{I}$ to retailer $j \in \mathcal{I}$, for $n \in \mathcal{N}^t$, $t \in \mathcal{T}''$;

$I_i^n \in \mathbb{Z}$: inventory levels at retailer $i \in \mathcal{I}$, for $n \in \mathcal{N}^t$, $t \in \mathcal{T}''$.

The multi-stage mixed integer non-linear programming model (MINLP) is formulated as follows:

Problem $\mathcal{P}1$

$$\begin{aligned}\min \ & \sum_{t\in\mathcal{T}'}\sum_{n\in\mathcal{N}^t}\pi^n\Big[\sum_{i\in\mathcal{I}}(f_i x_i^n + F_i v_i^n)\Big] + \\ & + \sum_{t\in\mathcal{T}''}\sum_{n\in\mathcal{N}^t}\pi^n\Big[\sum_{i\in\mathcal{I}}(g_i b_i^n + G_i r_i^n) + \sum_{i\in\mathcal{I}} h_i \max\{I_i^n, 0\} + \\ & + \sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{I}:i\neq j}(t_{ij} y_{ij}^n + T_{ij} V_{ij}^n) - \sum_{i\in\mathcal{I}} p_i \min\{I_i^n, 0\}\Big] \\ s.t.\end{aligned} \tag{2.1}$$

$$I_i^n = \max\{I_i^{a(n)}, 0\} + x_i^{a(n)} + b_i^n - d_i^n + \sum_{j\in\mathcal{I}:i\neq j}(y_{ji}^n - y_{ij}^n) \qquad i \in \mathcal{I}, n \in \mathcal{N}^t,\ t \in \mathcal{T}'' \tag{2.2}$$

$$\max\{I_i^{a(n)}, 0\} + x_i^{a(n)} + b_i^n + \sum_{j\in\mathcal{I}:j\neq i} y_{ji}^n \leq Q_i \qquad i \in \mathcal{I}, n \in \mathcal{N}^t,\ t \in \mathcal{T}'' \tag{2.3}$$

$$x_i^n \leq C v_i^n \qquad i \in \mathcal{I}, n \in \mathcal{N}^t,\ t \in \mathcal{T}' \tag{2.4}$$

$$b_i^n \leq C^{TB} r_i^n \qquad i \in \mathcal{I}, n \in \mathcal{N}^t,\ t \in \mathcal{T}'' \tag{2.5}$$

$$y_{ij}^n \leq C^{TB} V_{ij}^n \qquad i \in \mathcal{I},\ j \in \mathcal{I} : j \neq i, n \in \mathcal{N}^t,\ t \in \mathcal{T}'' \tag{2.6}$$

$$\sum_{j\in\mathcal{I}:j\neq i} y_{ij}^n \leq \max\{0, (I_i^{a(n)} + x_i^{a(n)} - d_i^n)\} \qquad i \in \mathcal{I}, n \in \mathcal{N}^t,\ t \in \mathcal{T}'' \tag{2.7}$$

$$x_i^n \geq 0 \ integer \qquad i \in \mathcal{I},\ n \in \mathcal{N}^t,\ t \in \mathcal{T}' \tag{2.8}$$

$$v_i^n \geq 0 \ integer \qquad i \in \mathcal{I},\ n \in \mathcal{N}^t,\ t \in \mathcal{T}' \tag{2.9}$$

$$y_{ij}^n \geq 0 \ integer \qquad i \in \mathcal{I},\ j \in \mathcal{I} : j \neq i,\ n \in \mathcal{N}^t,\ t \in \mathcal{T}'' \tag{2.10}$$

$$V_{ij}^n \geq 0 \; integer \qquad i \in \mathcal{I},\; j \in \mathcal{I} : j \neq i,\; n \in \mathcal{N}^t,\; t \in \mathcal{T}'' \tag{2.11}$$

$$b_i^n \geq 0 \; integer \qquad i \in \mathcal{I},\; n \in \mathcal{N}^t,\; t \in \mathcal{T}'' \tag{2.12}$$

$$r_i^n \geq 0 \; integer \qquad i \in \mathcal{I},\; n \in \mathcal{N}^t,\; t \in \mathcal{T}'' \tag{2.13}$$

$$I_i^n \; free \; and \; integer \qquad i \in \mathcal{I},\; n \in \mathcal{N}^t,\; t \in \mathcal{T}'' \tag{2.14}$$

The first sum in the objective function (2.1) denotes the expected retailers' variable and fixed shipment costs, while the second represents the expected backordering, inventory, transshipment and penalty costs.

Constraints (2.2) represent the inventory levels at the end of each stage $t \in \mathcal{T}''$, while Constraints (2.3) imply that the maximum inventory level at retailer i cannot exceed the warehouse capacity Q_i for each stage $t \in \mathcal{T}''$. Constraints (2.4), (2.5) and (2.6) link together the decision variables x_i^n, b_i^n and y_{ij}^n with the respective integer variables v_i^n, r_i^n and V_{ij}^n. Moreover, constraints (2.7) impose that the total transshipped quantities from retailer i cannot be greater than the sum of the total available quantity and the quantity received by direct shipping, minus the demand. Finally, constraints from (2.8) to (2.14) are variables definition constraints.

Due to the non-linearity of Problem $\mathcal{P}1$ for the presence of max and min terms, we linearize it following the approach described in Cavagnini, Bertazzi, Maggioni, and Hewitt [19] (2018) and we call the linearized problem "Problem $\mathcal{P}1^L$" (see Appendix 1.a for details). Without loss of generality, in what follows, we will sometimes refer to Problem $\mathcal{P}1^L$ as "Case (1)".

2.4 Two particular cases

In this section, we study in more detail two particular cases in order to obtain their optimal solutions analytically and to provide some complexity results.

Case (2): only external backordering (uncapacitated)

This case arises at the operational level, when a supplier has to ship an unknown demand to retailers (with unlimited warehouses capacity) only through backordering by using external transportation companies, so that vehicles fixed costs and capacities can be ignored.

Consequently, the model is the same of Problem $\mathcal{P}1^L$, with the exception that the decision variables concerning direct shipment and transshipment are removed and constraints (2.3) and (2.5) are not binding. Then, the model reduces to:

Problem $\mathcal{P}2$

$$\min \sum_{t\in\mathcal{T}''}\sum_{n\in\mathcal{N}^t}\pi^n\Big[\sum_{i\in\mathcal{I}}g_i b_i^n+\sum_{i\in\mathcal{I}}h_i\max\{I_i^n,0\}-\sum_{i\in\mathcal{I}}p_i\min\{I_i^n,0\}\Big] \tag{2.15}$$

s.t.

$$I_i^n=\max\{I_i^{a(n)},0\}+b_i^n-d_i^n \qquad i\in\mathcal{I},n\in\mathcal{N}^t,\ t\in\mathcal{T}'' \tag{2.16}$$

$$b_i^n\geq 0\ integer \qquad i\in\mathcal{I},\ n\in\mathcal{N}^t,\ t\in\mathcal{T}'' \tag{2.17}$$

$$I_i^n\ free\ and\ integer \qquad i\in\mathcal{I},\ n\in\mathcal{N}^t,\ t\in\mathcal{T}''. \tag{2.18}$$

Proposition 2.4.1. *The optimal total cost of Problem $\mathcal{P}2$ is*

$$z^*=\min\{\sum_{t\in\mathcal{T}''}\sum_{n\in\mathcal{N}^t}\pi^n\sum_{i\in\mathcal{I}}g_i(d_i^n-I_i^{a(n)})\}, \tag{2.19}$$

and, $b_i^{n*}=d_i^n-I_i^{a(n)}, i\in\mathcal{I},n\in\mathcal{N}^t,t\in\mathcal{T}^{''}$ and $x_i^{n*}=0, i\in\mathcal{I},n\in\mathcal{N}^t,t\in\mathcal{T}^{'}$. Therefore, an optimal policy can be computed in $O(\sum_{t=1}^{|\mathcal{T}''|}\mid\mathcal{N}^t\mid\cdot\mid\mathcal{I}\mid)$ time.

Proof. As for the company it is not convenient to pay inventory or stockout costs, from constraint (2.16), it is clear that the optimal solution is the one such that I_i^{n*}=0. Consequently, the optimal solution should be the one according to which: $b_i^{n*}=d_i^n-I_i^{a(n)}$, and, substituting in the objective function (2.15), we obtain the optimal total cost. □

Since this stochastic program has fixed objective coefficients, fixed recourse matrix and the uncertainty appears in the right-hand-side of the model, we know that the chain $EV\leq WS\leq RP\leq EEV$ (see Birge and Louveaux [12] (2011)) holds and, furthermore, for case (2), the chain is valid with equalities: $EV=WS=RP=EEV$. Specifically, we note that $EV=RP$ because the optimal first-stage solution is equal zero for both the EV and the RP solution, i.e. $x_i^{n*}=\overline{x}_i^{n*}=0$. Consequently, $EV=RP$ and $EEV=RP$ because the first-stage solution is again $\overline{x}_i^{n*}=0$.

Case (3): under-sized warehouses (capacitated)

Assume that $d_i^n>Q_i, i\in\mathcal{I},n\in\mathcal{N}^t,t\in\mathcal{T}^{''}$, that is the demand is always greater than the retailers' warehouse capacity, and that only direct shipments are allowed. Then, the model reduces to:

Problem $\mathcal{P}3$

$$\min \sum_{t\in\mathcal{T}'}\sum_{n\in\mathcal{N}^t}\pi^n\Big[\sum_{i\in\mathcal{I}}(f_i x_i^n+F_i v_i^n)\Big]+\sum_{t\in\mathcal{T}''}\sum_{n\in\mathcal{N}^t}\pi^n\Big[-\sum_{j\in\mathcal{I}}p_j\min\{I_j^n,0\}\Big] \tag{2.20}$$

s.t.

$$I_i^n=\max\{I_i^{a(n)},0\}+x_i^{a(n)}-d_i^n \qquad i\in\mathcal{I},n\in\mathcal{N}^t,\ t\in\mathcal{T}'' \tag{2.21}$$

$$\max\{I_i^{a(n)},0\}+x_i^{a(n)}\leq Q_i \qquad i\in\mathcal{I},n\in\mathcal{N}^t,\ t\in\mathcal{T}'' \tag{2.22}$$

$$x_i^n \leq Cv_i^n \qquad i \in \mathcal{I}, n \in \mathcal{N}^t,\ t \in \mathcal{T}' \tag{2.23}$$

$$x_i^n \geq 0\ integer \qquad i \in \mathcal{I},\ n \in \mathcal{N}^t,\ t \in \mathcal{T}'' \tag{2.24}$$

$$v_i^n \geq 0\ integer \qquad i \in \mathcal{I},\ n \in \mathcal{N}^t,\ t \in \mathcal{T}' \tag{2.25}$$

$$I_i^n\ free\ and\ integer \qquad i \in \mathcal{I},\ n \in \mathcal{N}^t,\ t \in \mathcal{T}''. \tag{2.26}$$

In fact, since $d_i^n > Q_i$, there are no surplus quantities and inventory costs does not occur.
Before introducing the following result, we specify that, the shipment cost for a *full-container-load (FCL)* per unit (FCLUC) is represented by $f_i + \frac{F_i}{C}$. On the other hand, the shipment cost for a *less-container-load (LCL)* per unit (LCLUC) is represented by $f_i + \frac{F_i}{Q_i - I_i^{a(n)} - C\lfloor \frac{Q_i - I_i^{a(n)}}{C} \rfloor}$.

Proposition 2.4.2. *For Problem $\mathcal{P}3$:*

a) If $FCLUC < p_i$ and $LCLUC < p_i$, the optimal policy is to deliver all the quantity in order to fill the warehouse (no matter il the vehicles are fully or partially loaded), i.e. $x_i^{n*} = Q_i - I_i^{a(n)}$, $v_i^{n*} = \lceil \frac{Q_i - I_i^{a(n)}}{C} \rceil$ and $I_i^{n*} = Q_i - d_i^n$. Hence, the optimal cost of Problem $\mathcal{P}3$ is:

$$z^* = \sum_{t \in \mathcal{T}'} \sum_{n \in \mathcal{N}^t} \pi^n [\sum_{i \in \mathcal{I}} f_i (Q_i - I_i^{a(n)}) + F_i \lceil \frac{Q_i - I_i^{a(n)}}{C} \rceil)] + \sum_{t \in \mathcal{T}''} \sum_{n \in \mathcal{N}^t} \pi^n [- \sum_{i \in \mathcal{I}} p_i (Q_i - d_i^n)]; \tag{2.27}$$

b) If $FCLUC < p_i$ and $LCLUC > p_i$, the optimal policy is to deliver only the quantity in order to completely fill the vehicles, i.e. $x_i^{n*} = C\lfloor \frac{(Q_i - I_i^{a(n)})}{C} \rfloor$, $v_i^{n*} = \lfloor \frac{Q_i - I_i^{a(n)}}{C} \rfloor$ and $I_i^{n*} = I_i^{a(n)} + C\lfloor \frac{Q_i - I_i^{a(n)}}{C} \rfloor - d_i^n$. Hence, the optimal cost of Problem $\mathcal{P}3$ is:

$$z^* = \sum_{t \in \mathcal{T}'} \sum_{n \in \mathcal{N}^t} \pi^n [\sum_{i \in \mathcal{I}} f_i C \lfloor \frac{(Q_i - I_i^{a(n)})}{C} \rfloor + F_i \lfloor \frac{Q_i - I_i^{a(n)}}{C} \rfloor)] + \sum_{t \in \mathcal{T}''} \sum_{n \in \mathcal{N}^t} \pi^n [- \sum_{i \in \mathcal{I}} p_i (I_i^{a(n)} + C \lfloor \frac{Q_i - I_i^{a(n)}}{C} \rfloor - d_i^n)]; \tag{2.28}$$

c) If $FCLUC > p_i$ and $LCLUC > p_i$, the optimal policy is to deliver zero quantities, i.e. $x_i^{n*} = 0$, $v_i^{n*} = 0$ and $I_i^{n*} = I_i^{a(n)} - d_i^n$. Hence, the optimal cost of Problem $\mathcal{P}3$ is:

$$z^* = \sum_{t \in \mathcal{T}} \sum_{n \in \mathcal{N}} \pi^n [- \sum_{i \in \mathcal{I}} p_i (I_i^{a(n)} - d_i^n)]. \tag{2.29}$$

Therefore, an optimal policy can be computed in $O(\sum_{t=1}^{|\mathcal{T}|} \mid \mathcal{N}^t \mid \cdot \mid \mathcal{I} \mid)$ time.

Proof.

a) Assume, by contradiction, that $x_i^n \neq Q_i - I_i^a(n)$. Then two cases can arise:

- $x_i^n > Q_i - I_i^{a(n)}$: but this case cannot occur because of Constraints (2.22).

- $x_i^n < Q_i - I_i^{a(n)}$: in this case, the optimal cost of Problem $\mathcal{P}3$ can be written in the following way, where η represents the number of pallets such that $x_i^n + \eta = Q_i - I_i^{a(n)}$:

$$z^{*C} = \sum_{t\in\mathcal{T}} \sum_{n\in\mathcal{N}} \pi^n [\sum_{i\in\mathcal{I}} f_i(Q_i - I_i^{a(n)} - \eta) + F_i[\frac{Q_i - I_i^{a(n)} - \eta}{C}])] + \sum_{t\in\mathcal{T}} \sum_{n\in\mathcal{N}} \pi^n [-\sum_{i\in\mathcal{I}} p_i(I_i^{a(n)} + (Q_i - I_i^{a(n)} - \eta) - d_i^n)]. \tag{2.30}$$

  The variable shipment cost given by the first term in equation (2.30) is less than the variable shipment cost in equation (2.27), since it is proportional to the number of delivered pallets. Concerning the fixed cost, in equation (2.30), it is less or equal to the fixed cost in (2.27). As a consequence, the total shipping cost of equation (2.30) is lower than the total shipment cost of equation (2.27). Nevertheless, the stockout cost given by the second term in equation (2.30) is higher than the corrisponding term in equation (2.27) and, since $LCLUC < p_i$, the cost increase associated to each stockout unit is higher than the cost decrease associated to each shipped unit. Consequently, $z^* < z^{*C}$, contraddicting the assumption that z^{*C} is optimal.

b) - c) The proof works in a similar way of point a). □

2.5 Computational results

In this section, our aim is three-fold. First, for Cases (1), (2) and (3), we provide the maximum dimension of the multi-stage stochastic programming models (both polynomial and general cases) which can be solved by an off-the-shelf solver. Second, we want to test the sensitivity of the optimal solution with respect to (1) the considered time horizon, (2) the unit inventory cost, (3) the capacity of the vehicles used for shipment, transshipment and backordering, both in the case in which the company performs transshipment and backordering through owned vehicles, and in the case where the company outsources these operations. Third, we test the performance of the Rolling horizon heuristic, by considering different reduced time horizons and different inventory costs. With this analysis, we also want to quantify the *end-of-horizon* effect.

All codes were implemented in Python 3.6.1 and models were solved using Gurobi 7.5.1 solver. All computational experiments were run on an Intel Core i7-7500U 64-bit machine with 8 GB of RAM and 2.70 gigahertz processor.

We now describe the generated instances for all three considered cases.

2.5.1 Instances description

Our instances are inspired by a real company which sells sport equipments and clothing. We consider a time horizon of one working week (i.e. $\mathcal{T} = \{0, \ldots, 5\}, \mathcal{T}' = \{0, \ldots, 4\}, \mathcal{T}'' = \{1, \ldots, 5\}$). The company has one central depot with unlimited availability and two retailers (i.e. $\mid \mathcal{I} \mid$=2), corresponding to big stores.

The following instances have been generated for case (1). The capacity C of the vehicles used for shipment is equal to 33 pallets, while the one of the vehicles used for transshipment and backordering is 22 pallets. Furthermore, we define the value P of a pallet to be equal to 50 Euros, and, since the unit inventory costs approximatively correspond to 5% of the value of a pallet, we set the retailers' inventory costs h_i equal to 2.5 Euros. Since the penalty cost corresponds to a lost sale, we let p_j equal to $1P$, that is 50 Euros. The first retailer has a small capacity $Q_1 = 66$, while the second has a bigger capacity $Q_2 = 111$. The demand can only assume values multiples of 11, between 0 and 33. As such, the support of the demand probability distribution is in the set of the following integer numbers $\{0, 11, 22, 33\}$. Obviously, the retailer with bigger warehouse capacity is also characterized by higher demand values. Consequently, its probability distribution is given by a Beta distribution (α, β), where $\alpha = 17$ and $\beta = 10$, having average demand $\mathbb{E}(d) = 22.07172$ pallets. Conversely, the smaller retailer's demand follows a Beta distribution (α, β), where $\alpha = 6$ and $\beta = 10$, having average demand $\mathbb{E}(d) = 11.03636$ pallets. We analyze and compare two different options. The first one, which we call "A", refers to the situation in which the supplier executes shipments through owned vehicles, while transshipment and backordering are performed by using vehicles owned by an external company. In this case, for the traditional shipments, only fixed costs are considered (and, consequently, vehicle capacities), while for transshipment and backordering we only consider variable costs (and we neglect the capacities of vehicles used for transshipment and backordering). The second one, which we call "B", studies the situation in which shipments, transshipment and backordering are all executed by company owned vehicles. In this case, we will consider only fixed costs and vehicle capacities. In each of these two different options, we notice that Problem $\mathcal{P}1^L$ can be reduced into three special cases, which facilitate a trade-off analysis. In particular, we consider the following instances: (1) both transshipment and backordering are allowed; (2) transshipment is not allowed; (3) backordering is not allowed. Finally, we assume that performing backordering is more expensive than transshipment, since a longer distance has to be traveled, and that backordering is more expensive than direct shipping, as backordering is a recourse actions which requires extra workers' commitment and also subtracts vehicles availability for other operations. Table (2.1) summarizes the different instances.

For case (2), we set the capacity of the vehicles used for usual shipments and for transshipment and backordering equal to infinity, that is $C = +\infty$ and $C^{TB} = +\infty$, respectively. Furthermore, we set the

	Instance	f_i	F_i	t_{ij}	T_{ij}	g_i	G_i
Case A	**1A**	-	225	8.64	-	12.95	-
	2A	-	225	$+\infty$	-	12.95	-
	3A	-	225	8.64	-	$+\infty$	-
Case B	**1B**	-	225	-	190	-	285
	2B	-	225	-	$+\infty$	-	285
	3B	-	225	-	190	-	$+\infty$

Table 2.1: Shipment, transshipment and backordering fixed and variable costs for every case and instance.

inventory cost $h_i = 10$ Euros, the fixed shipment cost $F_i = 225$ Euros and the unit transshipment and backordering costs $t_{ij} = b_{ij} = 8.64$ Euros.

For case (3), we need to modify the support of the demand which is now {11, 22, 33, 44}. Moreover, we set the capacity of the vehicles used for usual shipments $C = 3$, the capacity of the vehicles used for transshipment and backordering $C^{TB} = 2$, and the retailers' warehouse capacities $Q_1 = 8, Q_2 = 10$, respectively. Finally, based on the three different subcases presented in Theorem 2.4.2, the fixed shipment cost, and the fixed transshipment and backordering costs must be modified as follows:

a) $F_i = 30, T_{ij} = G_i = 190$;

b) $F_i = 120, T_{ij} = G_i = 400$;

c) $F_i = 200, T_{ij} = G_i = 400$.

For all three cases, we represent the demand uncertainty through a complete scenario tree including six stages. Since we are considering two retailers, and four possible demand realizations for each retailer, the tree will have the following branching structure: 16 branches from the root, 16 branches from each of the second, third, fourth and fifth-stage nodes, resulting in $S = 16 \times 16 \times 16 \times 16 \times 16 = 1,048,576$ scenarios and 1,118,481 nodes.

2.5.2 Solving the multi-stage stochastic programming models

In this subsection, we illustrate statistics about the performance of GUROBI in solving cases (1)-(3). Specifically, Tables 2.2-2.4 display the number of simplex iterations and the CPU time in seconds in the cases (1), (2) and (3), respectively, required by GUROBI to find an optimal solution of the stochastic programming model, for an increasing number of stages. We notice that, as our model is a MILP, the

simplex iterations refer to the total number of iterations done by the simplex algorithm in all the steps of the branch and bound relaxations.

These results show that GUROBI runs out of memory starting from the sixth-stage, even for the polynomially solvable cases. This makes the optimal policies provided in Section 2.4 even more valuable. Furthermore, for case (1), an optimal solution can be obtained in a reasonable time up to the third-stage, while with four stages (for some instances) and with five stages (for all instances), GUROBI is not able to provide an optimal solution with the time limit of 86400 seconds (24 hours). Furthermore, we observe that instances characterized by only fixed costs for transshipment and backordering are the most difficult to solve. Consequently, heuristic algorithms, like the Rolling horizon approach, are required.

			Two-stage	Three-stage	Four-stage	Five-stage	Six-stage
		Number of scenarios	16	256	4096	65536	1048576
		Number of nodes	17	273	4369	69905	1118481
Case A	1A	Simplex iterations	136	12355	4616013	32752088	Out of memory
		CPU time (seconds)	0.05	2.82	7614.90	86400 (11.81% gap)	-
	2A	Simplex iterations	28	904	55551	57055701	Out of memory
		CPU time (seconds)	0.04	0.33	38.93	86400 (0.06% gap)	-
	3A	Simplex iterations	331	24835	167681654	8373595	Out of memory
		CPU time (seconds)	0.11	3.84	86400 (8.98e-08% gap)	86400 (37.70% gap)	-
Case B	1B	Simplex iterations	686	169068	178691772	10349728	Out of memory
		CPU time (seconds)	0.20	17.99	86400 (0.72% gap)	86400 (19.44%)	-
	2B	Simplex iterations	58	1911	916731	55073892	Out of memory
		CPU time (seconds)	0.04	0.61	434.23	86400 (1.52% gap)	-
	3B	Simplex iterations	761	60072	73834968	7865480	Out of memory
		CPU time (seconds)	0.19	7.88	86400 (0.27% gap)	86400 (44.10% gap)	-

Table 2.2: Case (1): Summary statistics

	Two-stage	Three-stage	Four-stage	Five-stage	Six-stage
Simplex iterations	0	0	0	0	Out of memory
CPU time (seconds)	0.01	0.10	1.83	32.84	-

Table 2.3: Case (2): Summary statistics

		Two-stage	Three-stage	Four-stage	Five-stage	Six-stage
a)	Simplex iterations	1	1	1	1	Out of memory
	CPU time (seconds)	0.03	0.06	0.68	13.82	-
b)	Simplex iterations	1	1	4	3	Out of memory
	CPU time (seconds)	0.04	0.09	0.86	16.58	-
c)	Simplex iterations	2	2	6	7	Out of memory
	CPU time (seconds)	0.03	0.07	0.67	12.71	-

Table 2.4: Case (3): Summary statistics for subcases a), b), c)

2.5.3 Analysis of the (T+1)-stage stochastic model

In this subsection, we perform a sensitivity analysis of the optimal solution and of the total cost for case (1) with respect to the reduced time horizon T. Our aim is to understand if different values of the unit inventory cost, of the reduced time horizon, and of different vehicles capacities affect the optimal value of the first-stage variable and the total cost. We focus only on case (1) for two reasons. The first is that case (2) and case (3) are subject to particular parameter requirements which may be violated by performing this sensitivity analysis. The second is that these cases may not depend on the modified parameters (for example, the optimal solution of case (3) does not depend on the vehicle capacities, but on the warehouse capacities).

Table 2.5 shows the optimal value of the first-stage variables x^{0*} and the total cost of case (1) for different time horizons, different inventory unit costs (5% or 20% of the item unit price P) and different vehicle capacities. We do not show the optimal values of x^{n*} for $n \in \mathcal{N}^t,\ t > 0$, because they depend on the nodes. Furthermore, we only considered $T = 1$ (two-stage), $T = 2$ (three-stage), as the optimal solution cannot be computed in reasonable time for larger T. In the following paragraphs, we present insights which can be drawn from Table 2.5.

Recourse actions analysis

From the results, we observe that, indipendently of the time horizon length, of the unit inventory cost and of the vehicle capacities, it is more convenient to outsource the operations of backordering and transshipment. This is due to the fact that, in Case A (where backordering and transshipment are outsourced), there is more freedom in the quantities which are moved through backordering and transshipment, since only variable costs are applied. Conversely, in Case B (where backordering and transshipment are performed by the company through owned vehicles), fixed costs are paid and they impact more if the vehicles capacities are not fully utilized. This also explains the higher number of

		T=1 (two-stage)				T=2 (three-stage)			
		$C = 33, C^{TB} = 22$		$C = 22, C^{TB} = 11$		$C = 33, C^{TB} = 22$		$C = 22, C^{TB} = 11$	
Instance	h_i	x^{0*}	Total cost	x^{0*}	Total cost	x^{0*}	Total cost	x^{0*}	Total cost
1A	5%	[0, 33]	336.29	[0, 22]	377.80	[33, 33]	656.95	[0, 22]	753.27
	20%	[0, 33]	353.33	[0, 22]	378.84	[0, 33]	691.99	[0,22]	756.51
2A	5%	[0, 22]	383.18	[0, 22]	383.18	[0, 33]	679.54	[0, 22]	758.19
	20%	[0, 22]	390.14	[0, 22]	390.14	[0, 33]	762.73	[0, 22]	771.52
3A	5%	[0, 33]	424.45	[22, 22]	492.91	[33, 33]	727.39	[22, 22]	857.53
	20%	[0, 33]	441.49	[22, 22]	575.79	[0, 33]	881.11	[22, 22]	982.69
1B	5%	[0, 33]	415.23	[22, 22]	496.41	[33, 33]	737.36	[22, 22]	825.85
	20%	[0, 33]	442.48	[0, 22]	531.39	[0, 33]	874.00	[22, 22]	942.41
2B	5%	[0, 33]	495.26	[22, 22]	505.87	[33, 33]	800.44	[22, 22]	834.38
	20%	[0, 22]	503.11	[11, 22]	544.72	[22, 22]	949.77	[22, 22]	956.26
3B	5%	[0, 33]	496.77	[22, 22]	500.24	[33, 33]	782.06	[22, 22]	871.37
	20%	[0, 33]	513.80	[22, 22]	583.12	[22, 22]	981.08	[22, 22]	996.52

Table 2.5: Case (1): Optimal value of the first-stage variables x^{0*} and of the total cost, for different time horizons, different unit inventory costs and different vehicle capacities.

delivered pallets in cases B. Furthermore, both in Case A and B, lower costs are obtained if both backordering and transshipment are allowed, followed, in general, by the cases in which only backordering can be performed as recourse action. As a matter of fact, backordering gives a higher degree of flexibility with respect to transshipment, since new quantities can be introduced in the distribution system. Finally, we observe that the main difference between Cases A and B is given by the instances in which backordering is allowed. This suggests that when transshipment can be performed, the capacity of the vehicles used for transshipment is fully or almost fully used. On the contrary, when backordering can occur, the capacity of the vehicles used for backordering is only partially used.

Time horizon length impact

From Table 2.5, we observe that, as expected, as the time horizon length increases, the costs increase too. However, the increase is less than proportional, suggesting that a decision-maker would be able to take a better decision by considering a longer time horizon. At the same time, also the delivered quantities in the first-stage increase as the time horizon length increases, and in particular in the case with higher vehicles capacity and if the company performs all the operations using owned vehicles. This increase is due to the *end-of-horizon effect*.

Furthermore, when the unit inventory costs and the vehicles capacities are low, there is no difference

in the first-stage variables solution. With low capacities and high unit inventory costs, we only notice a difference for the case in which the company performs backordering by using its own vehicles (cases 1B and 2B). As a matter of fact, for these cases, as the time horizon length increases, the optimal first-stage variables solution suggests to deliver a higher number of pallets (equal to the capacity of the vehicle used for usual deliveries). If the capacities are high, the optimal first-stage solution suggests to deliver more pallets as the time horizon length increases. In particular, more quantities are delivered if the inventory costs are low, and the increase is higher if the company performs all operations with owned vehicles in order to utilize the capacities more.

Inventory unit costs impact

From Table 2.5, we observe that, in general, by increasing the inventory costs, the delivered quantities are the same or decrease. In particular, focusing on the two-stages case, no matter the vehicle capacity, these decreases incur when the company performs the operations by itself, and, in particular only in the cases where backordering is admitted. In fact, since backordering allows to introduce more quantities in the distribution system, when the inventory costs are high, the optimal solution would suggest to deliver fewer quantities in the first stage (so that the delivery is postponed after demand realization). Considering the three-stages results, if vehicles capacities are low, there will be no changes in delivered quantities, also by varying inventory costs. On the other hand, with higher capacities, the solution changes, especially for the cases in which transshipment is allowed. This suggests that transshipment occurs more often in these cases.

Vehicles capacity impact

We now analyze the impact of reduced vehicle capacities on the optimal solution. In general, we observe that, with a reduced capacity, the costs are higher, as more vehicles are needed and, thus, more fixed costs must be paid. In the second-stage case, the increase in costs can be observed especially for cases with higher inventory costs, if the company performs the recourse actions with owned vehicles, and if transshipment is allowed. In the three-stage case, the reduced vehicles capacity causes higher costs especially for low inventory costs (as, potentially, more quantities can be delivered) and if the company outsources the backordering and transshipment operations, especially for cases in which transshipment is allowed.

2.5.4 The Rolling horizon approach

Due to the impossibility of obtaining an optimal solution in reasonable time for some instances of case (1), starting from the fourth-stage, heuristic algorithms could be valuable and required. As such, in

this subsection, we analyze the performance of the Rolling horizon approach.

The Rolling horizon approach consists in solving the multi-stage problem by building in sequence submodels with less number of consecutive periods. Specifically, at first, the $(W+1)$-stage stochastic programming model defined on $t = 0, 1, \ldots, W < T$ is optimally solved and only the values of the first-stage decision variables x_i^n, $i \in \mathcal{I}, n \in \mathcal{N}^t$, are stored. Then, the $(W+1)$-stage stochastic programming model defined on $t = 1, 2, \ldots, W+1$ is optimally solved by setting the initial inventory level equal to the residual quantity in retailers' warehouses at the end of the previous time period, represented by $I_i^{a(n)}$. Once again, only the first-stage variables values x_i^n, $i \in \mathcal{I}, n \in \mathcal{N}^t$ are captured. The process is repeated until stage $t = T - W$. After solving the last $(W+1)$-stage stochastic program, a W-stage stochastic program on $t = T - W + 1, T - W + 2, \ldots, T$ is solved and then a $(W-1)$-stage stochastic programming model on $t = T - W + 2, T - W + 3, \ldots, T$ is solved. The process is repeated until the 2-stage stochastic programming model defined on $t = T - 1, T$ is solved. As a result, the computational complexity of the problem decreases with respect to the recourse problem, since the number of scenarios is reduced by considering the stochasticity only one (or a few) stages at a time.

First, we test the performance of the Rolling horizon approach on all the instances of case (1), by considering $W = 1$ and $W = 2$ and a deadline $T = 3$ (that is, the four-stage model). We also led the analysis by considering $W = 1$ and $W = 2$ and a deadline $T = 4$ (i.e. the five-stage model). Nevertheless, for some subproblems of the case with $W = 2$, we were not able to obtain the optimal solution within the time limit of 86,400 seconds. Consequently, we report only the results for the case with $W = 1$.

Performance of the Rolling horizon heuristic with $T = 3$

In this subsection, we test the performance of the Rolling horizon approach on all the instances of case (1), by considering W=1 and W=2 and a deadline $T = 3$ (that is, the four-stage model).

We report the outcomes in Tables 2.6 and 2.7. From the results, we observe that by considering $W = 1$ the Rolling horizon heuristic performs poorly, with respect to the optimal policy obtained by solving the full stochastic program, with an average cost increase of 25.32% for instances solved to optimality, and an average cost increase of 30.75% for instances not solved to optimality within the time limit of 86,400 seconds. Instead, with $W = 2$, we obtain very good results in almost all the instances. Specifically, with respect to the instances for which the optimal solution has been obtained within the time limit of 86,400 seconds, the Rolling horizon approach gives the same solutions in almost all cases (Instances (1A), (2B)). Nevertheless, for instance (2A), we notice that the solution obtained through the Rolling horizon approach is worse than the optimal policy. In fact, the percent cost increase of the Rolling horizon approach with respect to the optimal cost is equal to 6.81%. Futhermore, we see that

with respect to instances which were not solved to optimality with the time limit of 86,400 seconds, the Rolling horizon approach provides the same solution quality (Instances(3A) and (3B))), or even a better solution (Instance (1B)). Finally, we notice that for all instances, the Rolling horizon approach with $W = 2$ is able to provide a good-quality solution in less time than the one required by the full stochastic program.

	$W = 1$			$W = 2$			Optimal policy		
Instance	x^{0*}	Total cost	CPU s	x^{0*}	Total cost	CPU s	x^{0*}	Total cost	CPU s
1A	[0, 33]	988.33	13.56	[33, 33]	919.45	132.55	[33, 33]	919.45	7614.90
2A	[0, 22]	1133.89	1.40	[0, 33]	991.55	11.12	[33, 33]	928.31	38.93
3A	[0, 33]	1232.07	21.58	[33, 33]	992.81	196.79	[33, 33]	992.81 (8.98e-08% gap)	86400
1B	[0, 33]	1231.41	129.94	[33, 33]	975.25	842.09	[33, 33]	975.27 (0.72% gap)	86400
2B	[0, 33]	1449.59	4.86	[33, 33]	990.72	16.14	[33, 33]	990.72	434.23
3B	[0, 33]	1429.69	102.78	[33, 33]	1007.65	240.32	[33, 33]	1007.65 (0.27% gap)	86400

Table 2.6: Optimal value of the first-stage variables x^{0*} and total cost and CPU time (in seconds) for the Rolling horizon approach with $W = 1$ and $W = 2$, respectively, for instances belonging to Case (1) with deadline $T = 3$ (i.e. four-stage) and unit inventory costs equal to 5% of the item price.

	$W = 1$		$W = 2$	
Instance	Total cost	CPU s	Total cost	CPU s
1A	7.49%	-99.82%	0.00%	-98.26%
2A	22.15%	-96,40%	6.81%	-71.44%
3A	24.10%	-99.98%	0.00%	-99.77%
1B	26.26%	-99.85%	-0.002%	-99.03%
2B	46.32%	-98.88%	0.00%	-96.28%
3B	41.88%	-99.88%	0.00%	-99.72%

Table 2.7: Comparison in terms of objective value and CPU seconds of the solution provided by the Rolling horizon approach, with $W = 1$ and $W = 2$, respectively, with respect to the optimal (or near-optimal) solution, for instances belonging to Case (1) with deadline $T = 3$ (i.e. four-stage) and unit inventory costs equal to 5% of the item price.

We now test how the Rolling horizon approach performs with another set of instances. Specifically, given the sensitivity results obtained in Table 2.5, we choose the set of instances with more variability in the solution with respect to the reduced time horizon, that is the set of instances for which the unit inventory costs are equal to 20% of the item price and the capacities of the vehicles used for direct shipment and for transshipment and/or backordering are the same than the ones of the previous test

(i.e. 33 and 22 pallets, respectively).

The results are shown in Tables 2.8 and 2.9. We observe that, by considering $W = 1$, the Rolling horizon heuristic provides worse-quality results than the ones provided by considering $W = 2$. In fact, by considering $W = 2$ and the instances which cannot be solved to optimality within the time limit of 86,400 seconds (Instances 3A, 1B, 2B, 3B), we always obtain better results (apart from instance 2B, in which only backordering is admitted). On the other hand, by considering the instances solved to optimality (Instances 1A and 2A), the solution provided by the Rolling horizon approach is very close to the optimal one (with the one of the case where only outsourced backordering is allowed as recourse action performing the worst).

	$W = 1$			$W = 2$			Optimal policy		
Instance	x^{0*}	Total cost	CPU s	x^{0*}	Total cost	CPU s	x^{0*}	Total cost	CPU s
1A	[0, 33]	1033.51	12.70	[0, 33]	1033.50	86.13	[0, 33]	1033.49	564.80
2A	[0, 22]	1153.64	1.42	[0, 33]	1150.52	3.90	[0, 22]	1145.59	12.92
3A	[0, 33]	1316.28	17.91	[0, 33]	1307.67	212.62	[0, 33]	1307.68 (0.09% gap)	86400
1B	[0, 33]	1322.12	144.84	[0, 33]	1294.49	2466.65	[0, 33]	1294.50 (0.24% gap)	86400
2B	[0, 22]	1509.33	2.10	[22, 22]	1394.32	18.67	[22, 22]	1394.31 (0.03% gap)	86400
3B	[0, 33]	1522.43	25.31	[22, 22]	1441.17	382.16	[22, 22]	1441.23 (0.50% gap)	86400

Table 2.8: Optimal value of the first-stage variables x^{0*} and total cost and CPU time (in seconds) for the Rolling horizon approach with $W = 1$ and $W = 2$, respectively, for instances belonging to Case (1) with deadline $T = 3$ (i.e. four-stage) and unit inventory costs equal to 20% of the item price.

	$W = 1$		$W = 2$	
Instance	Total cost	CPU s	Total cost	CPU s
1A	0.002%	-97.75%	0.001%	-84.75%
2A	0.70%	-89.00%	0.43%	-69.81%
3A	0.66%	-99.98%	-0.0008%	-99.75%
1B	2.13%	-99.83%	-0.0008%	-97.15%
2B	8.25%	-99.99%	0.0008%	-99.98%
3B	5.63%	-99.97%	-0.004%	-99.56%

Table 2.9: Comparison in terms of objective value and CPU seconds of the solution provided by the Rolling horizon approach, with $W = 1$ and $W = 2$, respectively, with respect to the optimal (or near-optimal) solution, for instances belonging to Case (1) with deadline $T = 3$ (i.e. four-stage) and unit inventory costs equal to 20% of the item price.

Figures (2.1) and (2.2) summarize the comparison between the objective function value of the optimal

(or near optimal) solution and the objective function value of the Rolling horizon approach, for every considered instance based on the different considered W. By considering $W = 1$ (see Figure (2.1)), we notice a poor performance, especially in the case with low inventory costs and in the cases where backordering and transshipment are performed by owned vehicles (Instances 1B, 2B, 3B). Nevertheless, from Figure (2.2), it is clear that, by considering $W = 2$, the Rolling horizon approach performs very well. In this case, the instances for which the Rolling horizon approach performs the worst are the ones in which only backordering is allowed as recourse action (in particular, if it is outsourced and if the unit inventory costs are low).

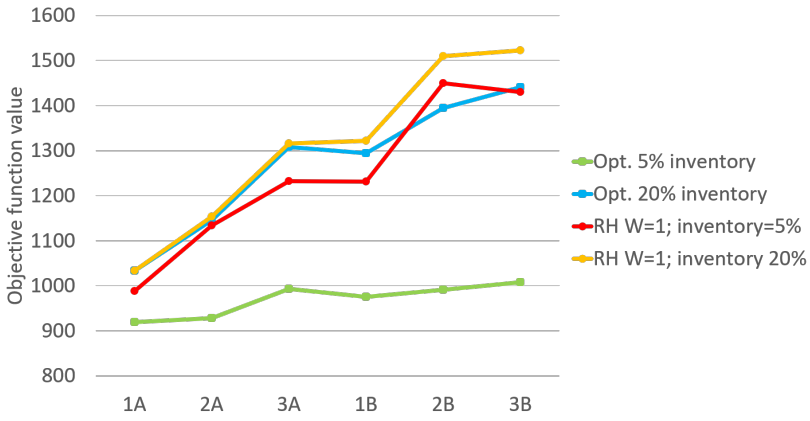


Figure 2.1: Total cost comparison between the optimal (or near optimal) solution and the Rolling horizon approach (RH) for every instance of Case (1) with $W = 1$ and deadline $T = 3$ and unit inventory costs equal to 5% and 20% of the unit item price.

To summarize, by considering $W = 2$, the *end-of-horizon effect* is mitigated (and in most instances, stability of the first-stage solutions is achieved). As a matter of fact, with $W = 2$, the solution provided by the Rolling horizon approach suggests to deliver an equal or higher amount of pallets than the one provided with $W = 1$. As a consequence, if a manager has to consider an horizon of 4 stages, it is sufficient to solve a three-stage stochastic program to end up with a good quality solution with low computational effort. Furthermore, considering $W = 2$ and instance 2A with low and high inventory costs, the *end-of-horizon effect* is only partially mitigated in the case of low inventory costs, as the discrepancy with respect to the quantities delivered in the optimal solution is higher. In fact, with low inventory costs, the Rolling horizon approach suggests to deliver 33 fewer pallets (in total) than the ones delivered in the optimal solution (which leads to lower warehouse availability at the end of the second stage). Conversely, with higher inventory costs, the Rolling horizon approach delivers 11 more pallets in total (which, in turn, leads to a superior warehouse availability at the end of the second stage). As such, the impact of the *end-of-horizon effect* caused by the Rolling horizon approach on

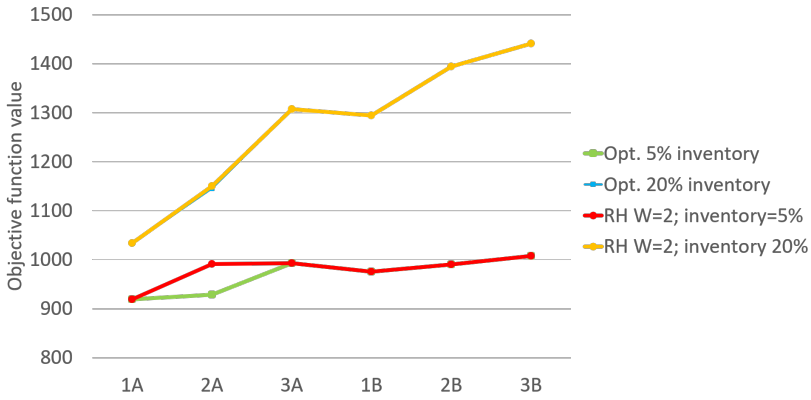


Figure 2.2: Total cost comparison between the optimal (or near optimal) solution and the Rolling horizon approach (RH) for every instance of Case (1) with $W = 2$ and deadline $T = 3$ and unit inventory costs equal to 5% and 20% of the unit item price.

total costs is different. In the first case (with low inventory costs), there is more need for backordering operations or more stockout, while in the second case (with high inventory costs), the inventory costs will be higher.

Performance of the Rolling horizon approach with $T = 4$

In this subsection, we test the performance of the Rolling horizon approach on all the instances of case (1), by considering W=1 and a deadline $T = 4$ (that is, the five-stage model).
We report the outcomes in Table 2.10. From the results, we can see that the Rolling horizon approach provides a better solution than the one of the recourse problem, only in three cases out of six, and, specifically, for those in which transshipment is outsourced and allowed, and for the one with transshipment as unique recourse action performed by owned vehicles. Consequently, it cannot be used as a valid tool for solving the five-stage model and alternative methods will be evaluated for future developments.

2.6 Managerial insights

In this section, we summarize the managerial insights which can be drawn from our computational experiments.
First, we have shown that case (1) is very difficult to be solved by the state-of-the-art solvers. For some instances belonging to this case, we are able to obtain the optimal solution in a time limit of 86,400 seconds, only up to three-stages. Thus, managers should consider the idea of using heuristics,

	$W = 1$			Optimal policy		
Instance	x	Total cost	CPU s	x	Total cost	CPU s
1A	[0, 33]	1315.02	5026.45	[0, 33]	1342.80 (11.81% gap)	86400
2A	[0, 22]	1508.93	34.3	[33, 33]	1247.19 (0.06% gap)	86400
3A	[0, 33]	1621.59	8330.65	[22, 22]	1828.57 (37.70% gap)	86400
1B	[0, 33]	1642.69	18246.75	[22, 33]	1479.94 (19.44% gap)	86400
2B	[0, 33]	1927.41	266.43	[33, 33]	1348.11 (1.52% gap)	86400
3B	[0, 33]	1880.19	11748.6	[11, 33]	2026.69 (44.10% gap)	86400

Table 2.10: Optimal value of the first-stage variables x^{0*} and total cost and CPU time (in seconds) for the Rolling horizon approach with $W = 1$, for instances belonging to case (1) with deadline $T = 4$ (i.e. five-stage model) and unit inventory costs equal to 5% of the item price.

	$W = 1$	
Instance	Total cost	CPU s
1A	-2.07%	-94.18%
2A	20.99%	-99.96%
3A	-11.32%	-90.36%
1B	11.00%	-78.88%
2B	42.97%	-99.69%
3B	-7.23%	-86.40%

Table 2.11: Comparison in terms of objective value and CPU seconds of the solution provided by the Rolling horizon approach, with $W = 1$, with respect to the optimal (or near-optimal) solution, for instances belonging to Case (1) with deadline $T = 4$ (i.e. five-stage) and unit inventory costs equal to 5% of the item price.

such as the Rolling horizon approach.

As a consequence, the performance of the Rolling horizon approach has been tested. The results show that, in general, by properly decomposing the time horizon on which the stochastic program has to be repeatedly solved, managers can get an advantage from using this heuristic approach both in terms of solution quality and of time. However, it is important for practitioners to carefully evaluate the average performance of the Rolling horizon approach in the particular set of instances which the company is interested to. As an example, in the case in which only outsourced backordering is allowed as recourse action, the performance of the Rolling horizon approach does not provide a good quality solution. Furthermore, the higher the inventory costs, the better the performance of the Rolling horizon approach is, as the delivered quantities are more stable.

Finally, through an extensive sensitivity analysis, we got multiple insights. Firstly, practitioners should rely both on transshipment and on backordering, and they should outsource these recourse operations

(especially in cases where backordering is allowed). Secondly, managers would be able to take a less myopic decision if they consider a longer time horizon (particularly, if the recourse actions are performed by owned vehicles with high capacities and the inventory costs are low). Thirdly, higher unit inventory costs impact more in the case in which transshipment is performed by the company through owned vehicles with high capacities. Finally, based on the choice of outsourcing the recourse operations or doing them internally, practitioners should carefully evaluate the vehicle capacities impact.

2.7 Conclusions

In this paper, we studied a distribution logistic system with transshipment and backordering with stochastic demand and proposed a multi-stage stochastic model. Two optimal policies for polynomially solvable cases have been presented and they turned out to be critical as GUROBI was able to provide a solution just up to the fifth-stage. Furthermore, since the general case cannot be solved to optimality in reasonable time starting from the fourth-stage, it is valuable to use heuristic methods. For this reason, we tested the performance of the Rolling horizon approach and we found out that, if the reduced time horizon is properly chosen, this heuristic provides good quality results with a very reduced computational effort. Finally, through a sensitivity analysis, we drew managerial insights concerning the time horizon, the unit inventory costs and the vehicle capacities influence on the optimal solution. Future developments of this work include a worst-case analysis of the Rolling horizon approach applied to this problem. Finally, considering that every subproblem of the Rolling horizon approach can be solved to optimality within the time limit of 86,400 seconds only up to the four-stage model, the performance of this approach under time limits for the five-stage model will be investigated and compared to the solution obtained by solving the full stochastic program.

Chapter 3

A two-stage stochastic optimization model for the Bike sharing allocation and rebalancing problem

Authors: Rossana Cavagnini[1], Luca Bertazzi[2], Francesca Maggioni[3] and Mike Hewitt[4]

(An extract of this chapter is under evaluation in Omega. Manuscript Reference Number: OMEGA_2018_539)

Keywords: Bike sharing, Rebalancing, Stochastic programming, Stochastic solution analysis.

[1]University of Bergamo, Via dei Caniana, 2, Bergamo, Italy, e-mail: r.cavagnini@studenti.unibg.it
[2]University of Brescia, Contrada Santa Chiara, 50, Brescia, Italy, e-mail: luca.bertazzi@unibs.it
[3]University of Bergamo, Via dei Caniana, 2, Bergamo, Italy, e-mail: francesca.maggioni@unibg.it
[4]Loyola University Chicago, Chicago, U.S.A, e-mail: mhewitt3@luc.edu

3.1 Introduction

Bike sharing systems are becoming more prevalent and popular throughout the world, doubling their number from 550 in 2012, to more than 1000 in 2016 (World Economic Forum [92] (2016)). In North America, bike sharing systems can be found in New York, Chicago, Washington, the San Francisco Bay Area, and Boston, whereas in South America, they can be found in Buenos Aires and Rio de Janeiro. In Europe, systems can be found in Copenhagen, London, and Paris. Finally, in Asia, the Wuhan and Hangzhou systems are two of the largest in the world, whereas in Oceania, large bike sharing systems can be found in Brisbane and Melbourne. The prevalence of these systems can be attributed to an increasing interest in reducing pollution and traffic, as well as promoting healthy lifestyles, worldwide. Bike sharing systems provide a fleet of bikes for use (typically via a rental agreement) by different individuals throughout the day. These systems typically consist of a depot (or set of depots), wherein bikes are stored at the beginning of the day, and multiple stations located throughout the city, from which an individual can withdraw a bike for a (usually short) journey and then return that bike when done. We note that the rider need not return the bike to the station from which it was withdrawn. These stations typically have a fixed number of slots for holding bikes, although their capacity can be expanded on a temporary basis. Finally, these stations often have technology that enables them to communicate information regarding their status (e.g. how many bikes are currently there) to a central manager/planner.

Typically, bike sharing systems are financed by public and/or private entities and managed by service providers, who are involved in strategic, tactical, and operational decision-making. Strategic decisions can include determining the number, location, and capacity of stations for bike rental and return, whereas tactical decisions can include fleet sizing and allocation decisions. Daily, operational decisions include determining how to periodically re-destribute bikes to stations.

This paper studies a bikesharing system composed of one depot (with an initial availability of bikes) and multiple capacitated stations. The capacity of the stations can be enlarged by individuals who can accept returning bikes even when the station is at capacity (the so called "valet service"). The service provider has to decide first how to allocate bikes to stations and then how to re-distribute bikes among stations, to rebalance the system. Rebalancing is performed after the realization of the demand, through a capacitated vehicle that follows a given route. The service provider tries to limit the cases in which an individual arrives at a station in hopes of renting a bike, but none is available (a situation we refer to as "*starvation*"). On the other hand, the service provider tries to limit the cases in which an individual seeks to return a bike to a station, but the station is already full (a situation we refer to as "*congestion*"). Both cases negatively impact the user's experience with the bike sharing service, as they both (potentially) require the user to travel to another station. At the same time, they

are somewhat competing objectives, as the more bikes allocated to a station, the less the likelihood of a starvation event, but the greater the likelihood of a congestion event. The service provider may also seek to limit the size of the bike fleet in use (to prevent it from demages and deterioration), as well as the number of bikes that are redistributed through the day.

We formulate a two-stage stochastic optimization model for allocating and re-distributing bikes, and show that explicitly acknowledging uncertainty leads to improvements along multiple performance dimensions over using a deterministic model. Specifically, our stochastic program includes objectives that measure each of the performance dimensions mentioned above (congestion, starvation, fleet size, rebalancing frequency). The impact of the stochastic demand on the problem solution is examined, showing the benefits of the proposed methodology in the solution quality when compared to solving the deterministic equivalent formulation. Nevertheless, by assessing the upgradeability of solutions from a deterministic version of this problem, we derive a heuristic procedure for solving the stochastic program that significantly reduces its solution time without losing solution quality.

To evaluate the plan produced by the stochastic program, we execute it in the context of a simulation of one week of operation for the San Francisco bike sharing system. To measure the impact of this plan on user experience, during this simulation we compute the frequency of congestion and starvation events at stations. To measure its impact on penalties, we measure the number of bikes allocated to and reallocated between stations. As the allocation of bikes to stations can be seen as an inventory problem, we propose heuristics based on a Newsvendor model-type analysis that differ in how the *critical ratio* is calculated. We see that the stochastic program is superior, as it outperforms both the Newsvendor heuristics and the actual plan on all dimensions.

The chapter is organized as follows. In Section 3.2, we discuss the relevant literature, and contrast both the problems studied and methodologies with the research proposed in this paper. Next, in Section 3.3, we describe the problem and, in Section 3.4, we formulate the associated model and we introduce the Newsvendor model based heuristics. In Section 3.5, we present computational results and analysis, while Section 3.6 includes managerial insights. Finally, Section 3.7 provides conclusions and suggestions for future works.

3.2 Literature Review

In this section, we review the literature that is relevant to the problem we study. It has five key features: (1) it includes various and contrasting objectives, as it considers both measures of customer service and the penalties associated with providing that service; (2) it is stochastic, as it explicitly recognizes that both the number of bikes rented from and returned to each station is not known with certainty but a

probability distribution is given; (3) it captures the opportunity to rebalance bikes at a point in time later in the day; (4) it considers that rebalancing is done with a fixed and static route, as motivated by the advantages highlighted in Erera, Savelsbergh, and Uyar [32] (2009), such as the regularity of the service, the simplification of the planning process and the familiarization of drivers with the delivery area. This is a fundamental difference from much of the existing literature, as much of it also considers the opportunity to determine the vehicle route used to support rebalancing each day; (5) it includes the presence of a Valet service, wherein station capacity is augmented by another resource (an individual) who can accept returning bikes. This service has been provided by the bikesharing system of Chicago (www.divvybikes.com/valet). When discussing the relevant literature, we will highlight how that work compares with respect to these features.

A natural categorization of the relevant literature is into problems that recognize uncertainty in bike usage and those that do not. Thus, in subsection 3.2.1 we first review the literature that considers deterministic problems and then, in subsection 3.2.2, the literature that studies problems that recognize this source of uncertainty. However, there is a literature on dynamic rebalancing problems, wherein (rebalancing) decisions are made while bikes are in use and thus, in subsection 3.2.3, we provide also a discussion on that literature.

3.2.1 Deterministic bikesharing problems

The majority of papers in the literature assumes that the use of bikes (i.e. how many are withdrawn from and returned to each station) is known when allocation and (potentially) rebalancing decisions are made. Raviv and Kolka [76] (2013) propose a deterministic bi-objective inventory model, in which the only decision to take is the initial allocation of bikes to stations. The model penalizes both when a user cannot withdraw a bike from a given station because it has stocked out as well as when a user cannot return a bike to a given station because it is at capacity. The model then seeks to minimize the sum of these penalties. However, these penalties are the same for all stations. In contrast, we propose to base these penalties on the distance between a station and the next closest station to capture that the individual may have to travel to that next closest station to either rent or return a bike. Differently from our work, this model also does not consider the option to rebalance at a later point in time.

Much of the literature focuses on what is referred to as the static bike rebalancing problem (BRP), which models the rebalancing of bikes at a point in time where there is very little usage (e.g. late at night). Such a problem can be viewed as a static pickup and delivery problem. A review and classification of this family of problems is presented in Berbeglia, Cordeau, Gribkovskaia, and Laporte [9] (2007). According to their classification, the problem we study is most similar to the one-to-one PDP (pick up and delivery problem) with transshipment, since each rebalancing flow has exactly one

pickup station and one delivery station, and these stations are determined by the order in which the rebalancing route visits stations. However, in contrast to our problem, this variant of the PDP does not model the customer service aspect of the problem.

Conversely, Raviv, Tzur, and Forma [77] (2013) model the static rebalancing problem with a multi-objective mathematical program wherein they seek to minimize an objective that consists of three terms: (1) penalties for stockouts, (2) penalties for stations being at capacity when users wish to return bikes, and, (3) the operational costs incurred when rebalancing. They present two formulations of the problem, with the first restricting how rebalancing can be performed in order to make the model easier to solve. The second formulation then relaxes some of those restrictions. A variant of the first formulation is also studied in Forma, Raviv, and Tzur [35] (2015) and in Ho and Szeto [45] (2014).

Benchimol, Benchimol, Chappert, De La Taille, Laroche, Meunier, and Robinet [8] (2011), Chemla, Meunier, and Calvo [22] (2013), and Erdoğan, Laporte, and Calvo [31] (2014) also focus on the static rebalancing problem. In addition, all three propose models that seek to minimize some measure of the costs incurred by rebalancing. Benchimol, Benchimol, Chappert, De La Taille, Laroche, Meunier, and Robinet [8] (2011) and Chemla, Meunier, and Calvo [22] (2013) model the desire to avoid starvation and congestion with the use of pre-determined target levels for the number of bikes at each station after rebalancing. Brinkmann, Ulmer, and Mattfeld [14] (2016) also consider a target level for the number of bikes at each station, but minimizes an objective that measures deviations from those targets. Similarly, Erdoğan, Laporte, and Calvo [31] (2014) presume a pre-determined range for how many bikes should be at each station, while we only impose a lower bound and similarly to our model, Erdoğan, Laporte, and Calvo [31] (2014) assume rebalancing is done with a single vehicle that follows a fixed route. Finally, Dell'Amico, Hadjicostantinou, Iori, and Novellani [28] (2014) and Dell'Amico, Iori, Novellani, Stützle, et al. [29] (2016) propose models that minimize rebalancing costs with the use of a fleet of capacitated vehicles, instead of a unique vehicle.

3.2.2 Stochastic bikesharing problems

We next turn our attention to models that recognize uncertainty in how bikes will be used. Lu [54] (2016) studies a problem that focuses on both the initial allocation and rebalancing of bikes, wherein there is uncertainty in how bike usage will deviate from what is expected. Their objective measures bike supply cost, inventory and redistribution costs, and penalties associated with stock-outs. Differently from our approach, they propose a robust model of this problem that seeks to minimize their objective under a maximum demand scenario generated from two different uncertainty sets. Brinkmann, Ulmer, and Mattfeld [13] (2015) present a stochastic model of the rebalancing problem in which they seek to minimize the expected number of starvations. They present heuristics for solving

this model. Conversely, Datner, Raviv, Tzur, and Chemla [27] (2017) focus solely on determining the initial allocation of bikes to stations, with the goal of minimizing the frequency of congestion and starvation events under demand realizations drawn from a stochastic process. However, in contrast to our model, they do not consider station capacities or that the fleet of bikes that may be allocated to stations can be of limited size.

Bike sharing systems are, in a sense, similar to car sharing systems. Fan [33] (2014) proposes a multistage stochastic model for allocating and redistributing cars wherein they consider uncertainty in the demand for cars at each "station". Their model seeks to maximize expected profit, which is calculated as the difference between total revenue and rebalancing costs. However, they do not consider the quality of service provided by the system (i.e. the number of starvation or congestion events) and they use only three scenarios, while our work presents an extensive study to better describe the uncertainty.

3.2.3 Dynamic rebalancing problem

In contrast to the static rebalancing problem, wherein it is assumed that rebalancing is done at night when no bikes are in use, researchers have also studied a dynamic rebalancing problem, wherein rebalancing is done while bikes are in use. As a result, in the dynamic problem, the locations of bikes may change during the rebalancing operation. For this problem, Ghosh, Varakantham, Adulyasak, and Jaillet [37] (2017) propose a deterministic optimization model that seeks to maximize a measure of profit that is a function of both user ride length and rebalancing costs. Finally, Regue and Recker [78] (2014), present a sequential framework for the dynamic rebalancing problem. The first step in this framework is to forecast demand at each station, from which a target inventory level for that station is determined. The second step is to determine a rebalancing plan based upon those target inventory levels, which dictates how many bikes should be transported from one station to another. Then, the third step is to determine vehicle routes to execute that rebalancing plan.

3.3 Problem description

In this section, we provide a description of the problem. Specifically, we study a bikesharing system managed by a service provider wherein the decision-making is centralized. The service provider has to determine the number of bikes to allocate at the beginning of the day and how they should be rebalanced at a later point in the day, under uncertain demand. When making these decisions, the service provider seeks to both ensure that a customer who seeks to rent a bike from a specific station can do so, and, that a customer who seeks to return a bike to a specific station can do so. In addition,

the service provider wishes to provide a high level of service with few bikes and little rebalancing. We consider a single depot (although the methods we propose could be extended to systems with multiple depots) and multiple stations, which are represented by the set $\mathcal{I}$. Since we presume rebalancing is done via a fixed route, $\mathcal{I}$ is an ordered set, with the ordering determined by the sequence in which the route visits stations. Since the route ends at the depot, we consider the depot as such a station, and represent it by I ($\in \mathcal{I}$). We assume that there is a limited number of bikes $\overline{I}_{I0}$ at the depot that can be allocated to stations and that this number also corresponds to the depot capacity. On the other hand, each station $i \in \mathcal{I} \setminus \{I\}$ is characterized by a bike capacity Q_i and by a number of bikes that is initially at that station, $\overline{I}_{i0}$ (which may be 0). We presume rebalancing is executed by a vehicle with capacity C, that travels along a known and fixed route (determined *a priori* by solving a Travelling Salesman Problem (TSP)) that begins at the depot, visits each station, and then ends at the depot. Finally, while each station has a capacity, it can also be expanded on a temporary basis using what has been referred to as a "valet service", which involves stations being staffed with individuals who can accept returning bikes even when the station is at capacity.

3.4 A two-stage stochastic programming formulation

In this section, we propose a two-stage stochastic programming formulation of the problem presented in Section 3.3. At the beginning of the day (say 6 a.m.), the first stage decision is to determine the number of bikes to deliver from the depot to each station $i \in \mathcal{I} \setminus \{I\}$ (before knowing the station demand) and we represent it with the variables x_i. In order to ensure that a station can satisfy bike rental requests in the early hours, before any bikes are returned, we compute a minimum number of bikes that have to be allocated to each station, $\underline{x}_i$ (which may be 0).

We presume that, when the decision regarding the allocation of bikes to stations is made, only a probability distribution of station demand is known. Recalling that the bike demand level at each station $i \in \mathcal{I} \setminus \{I\}$ is a random variable, we denote the set of possible realizations (or scenarios) of the uncertain parameter with $\mathcal{S}$. Furthermore, we represent the stochastic bike demand at station $i \in \mathcal{I} \setminus \{I\}$ in scenario $s \in \mathcal{S}$ with d_i^s and the probability assigned to each scenario $s \in \mathcal{S}$ with pr^s. Bike sharing systems are rental systems. Thus, product demand is different from what is considered in many inventory models wherein products are purchased, and from the supplier's perspective, disappear. As such, we measure demand at the station level, and during the period between when bikes are initially allocated and redistributed, as the difference between the number of rented and returned bikes at that station during that period. Note, this means that d_i^s may be either positive (more bikes withdrawn from station i than returned in scenario s) or negative (more bikes returned to station i than withdrawn

in scenario s).

Then, we presume that at a later point in the day (say 12 p.m.), after the system has been in use and consequently stations' demand has realized, the service provider observes the number of bikes at each station and determines a rebalancing plan. As a result, we model the determination of this rebalancing plan with second stage, scenario-dependent variables $y^s_{i,i+1}$, which dictate the number of bikes to re-distribute from station $i \in \mathcal{I} \setminus \{I\}$ to the next station on the fixed route. We do not currently model the opportunity to allocate bikes from the depot to a station during the rebalancing operation. However, the model can be extended to accommodate that operation. We notice that doing so allows us to measure the ideal fleet size as the total number of bikes initially allocated to stations. On the other hand, if we allowed rebalancing from the depot, the number of allocated bikes could not be representative of the fleet size, as it could be adjusted according to scenario realization.

We illustrate the sequence of decisions and events in Figure 3.1.

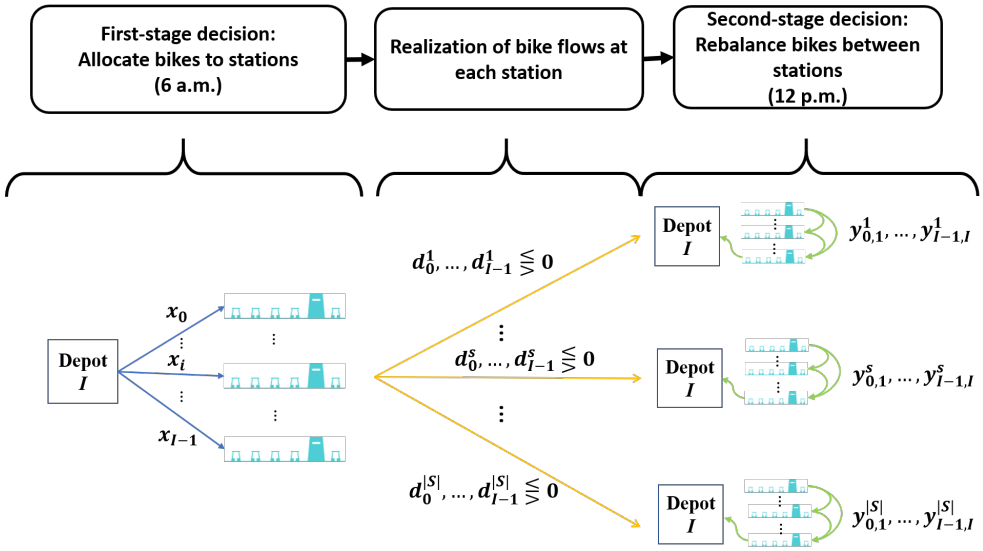


Figure 3.1: Illustration of the two-stage decision-making process

We recall that our problem includes different objectives. Specifically, the service provider seeks to avoid the occurrence of both congestion (a user wishes to return a bike to a station but it is full) and starvation (a user wishes to rent a bike from a station but it is empty). At the same time, the provider seeks to minimize both the number of bikes allocated to prevent bikes damage and rebalanced. To handle these different objectives, we next discuss how we measure each of these dimensions.

Starvation is measured by the number of bike rental requests at a station during the period of time between initial allocation and rebalancing that are in excess of the number of bikes that are positioned there at the end of the second stage, which we denote as I^{s-}_i, $i \in \mathcal{I} \setminus \{I\}$. We refer to the weight

associated with starvation as the "stock-out penalty" p_i, $i \in \mathcal{I} \setminus \{I\}$. Congestion is measured with two terms, B_i^{s+}, $i \in \mathcal{I} \setminus \{I\}$ and E_i^{s+}, $i \in \mathcal{I} \setminus \{I\}$, with the first measuring the number of bikes at a station above and beyond the number initially allocated (which we term "extra inventory"), and the second measuring the number of bikes in excess of station capacity (which we term "excess inventory"). We refer to the weight associated with "excess inventory" as the "excess penalty," $c_i, \quad \forall i \in \mathcal{I} \setminus \{I\}$ and the weight associated with "extra inventory" as the "extra penalty," $\frac{c}{Q}, \quad \forall i \in \mathcal{I} \setminus \{I\}$.

We choose to model both extra and excess inventory for multiple reasons. First, we interpret the first stage decisions as indicating the inventory level for bikes at each station that best prevents both congestion and starvation. Thus, we penalize when there are more bikes (but fewer that the station capacity) at a station than in that initial allocation. Second, as we seek to limit the occurrence of congestion, we penalize if the surplus with respect to the station capacity occurs, since it implies higher operational costs due for having a staff member at a station ("valet service").

Regarding the penalties associated with providing the service, the objective contains a term regarding initial allocation, which is measured as the number of bikes allocated to each station. A "delivery penalty", $f_i, \quad \forall i \in \mathcal{I} \setminus \{I\}$, is associated with this number. Finally, the objective includes a term regarding re-balancing, which is measured by the total number of bikes moved between stations. The weight associated with this quantity is referred to as the "rebalancing penalty" $t_{i,i+1}$, $i \in \mathcal{I} \setminus \{I\}$.

To address this problem, we now summarize the notation and we propose an integer non linear stochastic program, which we linearize (details in Appendix 2.a) in order to solve it with an off-the-shelf mixed integer programming (MIP) program solver.

Notation.

Sets:

$\mathcal{I} = \{1, \ldots, I\}$, set of stations (depot included);

$\mathcal{S} = \{1, \ldots, S\}$, set of scenarios;

Deterministic parameters:

$\underline{x}_i$, minimum number of bikes that has to be allocated to station $i \in \mathcal{I} \setminus \{I\}$;

$\overline{I}_{I0}$, initial availability of bikes at the depot and depot capacity;

Q_i, capacity of station $i \in \mathcal{I} \setminus \{I\}$;

$\overline{I}_{i0}$, initial availability of bikes at station $i \in \mathcal{I} \setminus \{I\}$;

C, capacity of the vehicle used for transshipment;

p_i, stock-out penalty at station $i \in \mathcal{I} \setminus \{I\}$;

c_i, excess penalty at station $i \in \mathcal{I} \setminus \{I\}$;

$\frac{c}{Q}$, extra penalty at station $i \in \mathcal{I} \setminus \{I\}$;

f_i, delivery penalty at station $i \in \mathcal{I} \setminus \{I\}$;

$t_{i,i+1}$, rebalancing penalty at station $i \in \mathcal{I} \setminus \{I\}$;

Stochastic parameters:

d_i^s, stochastic demand of bikes at station $i \in \mathcal{I} \setminus \{I\}$ in scenario $s \in \mathcal{S}$;

pr^s, probability of scenario $s \in \mathcal{S}$;

Variables:

$x_i \in \mathbb{Z}^+$, first-stage variables representing the number of bikes to allocate at station $i \in \mathcal{I} \setminus \{I\}$;

$y_{i,i+1}^s \in \mathbb{Z}^+$, second-stage variables representing the number of bikes to relocate from station $i \in \mathcal{I} \setminus \{I\}$ to station $i+1$;

$I_i^s \in \mathbb{Z}$, balance of bikes at station $i \in \mathcal{I} \setminus \{I\}$ in scenario $s \in \mathcal{S}$;

$I_i^{s+} \in \mathbb{Z}^+$, units of surplus at station $i \in \mathcal{I} \setminus \{I\}$ in scenario $s \in \mathcal{S}$;

$I_i^{s-} \in \mathbb{Z}^-$, units of stock-out at station $i \in \mathcal{I} \setminus \{I\}$ in scenario $s \in \mathcal{S}$;

$B_i^s \in \mathbb{Z}$, extra inventory balance at station $i \in \mathcal{I} \setminus \{I\}$ in scenario $s \in \mathcal{S}$;

$B_i^{s+} \in \mathbb{Z}^+$, units of extra inventory at station $i \in \mathcal{I} \setminus \{I\}$ in scenario $s \in \mathcal{S}$;

$E_i^s \in \mathbb{Z}$, excess inventory balance at station $i \in \mathcal{I} \setminus \{I\}$ in scenario $s \in \mathcal{S}$;

$E_i^{s+} \in \mathbb{Z}^+$, units of excess inventory at station $i \in \mathcal{I} \setminus \{I\}$ in scenario $s \in \mathcal{S}$;

The model is:

Problem $\mathcal{B}$

$$min \sum_{i \in \mathcal{I}} f_i x_i + \sum_{s \in \mathcal{S}} pr^s [\sum_{i \in \mathcal{I} \setminus \{I\}} (t_{i,i+1} y_{i,i+1}^s + \frac{c_i}{Q_i} B_i^{s+} + c_i E_i^{s+} + p_i(-I_i^{s-}))] \tag{3.1}$$

s.t:

$$x_i \geq \underline{x}_i \qquad i \in \mathcal{I} \setminus \{I\} \tag{3.2}$$

$$\overline{I}_{i0} + x_i \leq Q_i \qquad i \in \mathcal{I} \setminus \{I\} \tag{3.3}$$

$$\sum_{i \in \mathcal{I} \setminus \{I\}} x_i \leq \overline{I}_{I0} \tag{3.4}$$

$$y_{i,i+1}^s \leq C \qquad i \in \mathcal{I} \setminus \{I\},\ s \in \mathcal{S} \tag{3.5}$$

$$I_I^s = \overline{I}_{I0} - \sum_{i \in \mathcal{I} \setminus \{I\}} x_i + y_{I-1,I}^s \qquad s \in \mathcal{S} \tag{3.6}$$

$$I_I^s \leq \overline{I}_{I0} \qquad s \in \mathcal{S} \tag{3.7}$$

$$I_1^s = \overline{I}_{10} + x_1 - d_1^s - y_{1,2}^s \qquad s \in \mathcal{S} \tag{3.8}$$

$$I_i^s = \overline{I}_{i0} + x_i - d_i^s + y_{i-1,i}^s - y_{i,i+1}^s \qquad i \in \mathcal{I} \setminus \{1, I\},\ s \in \mathcal{S} \tag{3.9}$$

$$I_i^{s+} = \max\{0, I_i^s\} \qquad i \in \mathcal{I} \setminus \{I\},\ s \in \mathcal{S} \tag{3.10}$$

$$I_i^{s-} = \min\{0, I_i^s\} \qquad i \in \mathcal{I} \setminus \{I\},\ s \in \mathcal{S} \tag{3.11}$$

$$E_i^s = I_i^{s+} - Q_i \qquad i \in \mathcal{I} \setminus \{I\},\ s \in \mathcal{S} \tag{3.12}$$

$$E_i^{s+} = \max\{0, E_i^s\} \qquad i \in \mathcal{I} \setminus \{I\},\ s \in \mathcal{S} \tag{3.13}$$

$$B_i^s = I_i^{s+} - x_i - \overline{I}_{i0} - E_i^{s+} \qquad i \in \mathcal{I} \setminus \{I\},\ s \in \mathcal{S} \tag{3.14}$$

$$B_i^{s+} = \max\{0, B_i^s\} \qquad i \in \mathcal{I} \setminus \{I\},\ s \in \mathcal{S} \tag{3.15}$$

$$x_i \geq 0 \ \ integer \qquad i \in \mathcal{I} \setminus \{I\} \tag{3.16}$$

$$y_{i,i+1}^s \geq 0 \ \ integer \qquad i \in \mathcal{I} \setminus \{I\},\ s \in \mathcal{S} \tag{3.17}$$

$$I_i^s \ \ free\ and\ integer \qquad i \in \mathcal{I} \setminus \{I\},\ s \in \mathcal{S} \tag{3.18}$$

$$I_i^{s+} \geq 0 \ \ integer \qquad i \in \mathcal{I} \setminus \{I\},\ s \in \mathcal{S} \tag{3.19}$$

$$I_i^{s-} \leq 0 \ \ integer \qquad i \in \mathcal{I} \setminus \{I\},\ s \in \mathcal{S} \tag{3.20}$$

$$B_i^s \ \ free\ and\ integer \qquad i \in \mathcal{I} \setminus \{I\},\ s \in \mathcal{S} \tag{3.21}$$

$$B_i^{s+} \geq 0 \ \ integer \qquad i \in \mathcal{I} \setminus \{I\},\ s \in \mathcal{S} \tag{3.22}$$

$$E_i^s \ \ free\ and\ integer \qquad i \in \mathcal{I} \setminus \{I\},\ s \in \mathcal{S} \tag{3.23}$$

$$E_i^{s+} \geq 0 \ \ integer \qquad i \in \mathcal{I} \setminus \{I\},\ s \in \mathcal{S} \tag{3.24}$$

The objective function (3.1) represents the minimization of the total expected penalty, obtained through the sum of all the charged penalities for delivery, rebalancing, extra and excess inventory and stock-out. Constraints (3.2) impose that the delivered quantity to each station has to be at least as great as the initial requirement at that station. Constraints (3.3) guarantee that the sum between the quantity allocated and initially available at each station does not exceed the station capacity. Constraint (3.4) implies that the total number of delivered bikes to stations is less than the available quantity at the depot. Constraints (3.5) ensure that the number of bikes carried by the vehicle during rebalancing never exceeds its capacity in each scenario $s \in \mathcal{S}$. We recall that rebalancing occurs on a fixed route that begins and ends at the depot, but that rebalancing does not involve bringing bikes from the depot to the first station on this route. As such, constraints (3.6) ensure that, for the depot, in each scenario $s \in \mathcal{S}$, the quantity at the end of the period is equal to the initial bike availability and the quantity received from the last visited station minus the quantities delivered to stations. Constraints (3.7) ensure that, in each scenario $s \in \mathcal{S}$, at the end of the rebalancing period, the number of

bikes at the depot does not exceed its capacity. Moreover, the "flow balance" constraints for bikes at the first station on this route is different from the remaining stations. Specifically, constraints (3.8) ensure that, for the first visited station, the quantity at the end of rebalancing is equal to the sum between the initial available quantity and the quantity received from the depot minus the quantities used to satisfy the demand and those bikes that are redistributed to subsequent stations on the route in each scenario $s \in \mathcal{S}$. Similarly, constraints (3.9) determine the inventory position (which can be negative or positive) at a station other than the first, as a function of the initial inventory level, the number allocated, the number withdrawn/returned, and the number redistributed to another station in each scenario $s \in \mathcal{S}$. Constraints (3.10) and (3.11) determine the surplus and stock-out quantities, respectively, for each station and for each scenario $s \in \mathcal{S}$. Recalling that we model the presence of a valet service, wherein bikes can be returned to stations that are full, constraints (3.12) and (3.13) calculate the number of bikes at each station that are in excess of station capacity, in each scenario $s \in \mathcal{S}$. Similarly, constraints (3.14) and (3.15) determine, for each scenario $s \in \mathcal{S}$, when there are more bikes positioned at a station after rebalancing than were initially allocated, but not more than station capacity. If there is, these constraints ensure that B_i^{s+} represents that number of bikes. Finally, Constraints (3.16) to (3.24) are variable definition constraints.

We notice that, in this model, infeasibility cannot occur since there are no constraints imposing the demand satisfaction and if there are no bikes to satisfy the demand or if the station is more than full, a penalty is added in the objective function.

3.4.1 Newsvendor-based heuristics

The problem of determining the initial bikes allocation can be seen as an inventory model as we have penalties associated to inventory and stock-outs. Consequently, practitioners may be interested in assessing the quality of the solution if a Newsvendor model approach is used. In this section, we propose three heuristics for determining the initial allocation of bikes to stations. Each can be classified as a Newsvendor model-based approach (Cachon and Terwiesch [16] (2009)). Recall that the Newsvendor model is appropriate for a single-period inventory context wherein there is a penalty associated with stocking out (e.g. penalties associated with lost sales) that needs to be balanced against costs associated with having excess inventory (e.g. holding costs, salvage costs). It has been shown that there is a percentage of demand that should be served by inventory on-hand that maximizes profit. This percentage is often referred to as the *critical ratio*.

To use a Newsvendor (Cachon and Terwiesch [16] (2009)) approach, we calculate a critical ratio for each station that is a function of an underage component (the loss incurred with each stock-out, or, in our case, each customer that cannot rent a bike at a station when he/she wants to) and an overage

component (the loss incurred for each extra bike allocated to a station). Specifically, the critical ratio is calculated as $underage/(underage + overage)$. We call these "Sequence based heuristics (SBH)" and execute them by solving the stochastic program presented above, albeit with the values of the first-stage variables fixed to those indicated by the heuristic (the detailed algorithm is presented in Appendix 2.c). Consequently, the number of bikes to allocate to each station, which is determined by these Newsvendor approaches, is used as input in our original stochastic program, which returns the rebalancing decisions as output.

For all three heuristics, the underage is set to the "stock-out penalty", p_i, that is used in the objective function of the stochastic program. The heuristics differ in how the overage is calculated, which we discuss below:

- CR1: $CR_i = \frac{p}{p+c}$. In this case, the overage component accounts only for the penalty, c_i, associated with having bikes in excess of station capacity.
- CR2: $CR_i = \frac{p}{p+c+f+\cdot}$. In this case, the overage component includes the excess penalty, c_i, as well as the penalty, f_i, associated with delivering a bike to that station. However, the overage also includes the penalty c_i/Q_i incurred when there are "extra" bikes at a station relative to the initial allocation.
- CR3: $CR_i = \frac{p}{p+0.5c+0.5(f+\cdot)}$. The overage component again considers the cost of delivering a bike to a station. However, as the excess and extra penalties are not paid at the same time, the overage component is based on a 50% chance of each occurring.

Once the critical ratio is calculated, we determine the number of bikes to allocate. Specifically, for each station, we compute its cumulative probability distribution of the demand and, in order to establish the first stage variable, we select the demand level which corresponds to the value of the cumulative probability distribution equal to the critical ratio. Since we want to ensure that the minimum requirement $\underline{x}_i$ is met, letting $x_i(CR_i)$ represent the quantity to be allocated to station i that is prescribed by the critical ratio, we set the number of bikes to allocate to station i equal to $x_i^{NH} = \max(x_i(CR_i), \underline{x}_i)$. However, as the calculation of CR_i does not consider that there may be a limited number of bikes in total, $\overline{I}_{I0}$ (while the stochastic program recognizes this), we may have $\sum_{i\in\mathcal{I}\setminus\{I\}} x_i^{NH} > \overline{I}_{I0}$. When that is the case, we assume an ordering to the stations (derived from the order in which the vehicle visits the stations), and allocate bikes to stations in that order. Specifically, the heuristic delivers the determined quantity starting from the first visited station, and then continues on delivering to each subsequent station, until either there are no more bikes or each station has received its allocation. Notice that, coherently with the stochastic program, we do not assume a capacity for the vehicle used for allocating bikes to stations, so that the allocated bikes are not influenced by the

vehicle capacity. Similarly, these CR_i calculations do not consider station capacity. As such, if the newsvendor-derived quantity exceeds a station's capacity (i.e. $x_i^{NH} > Q_i$), the excess is delivered to a station that occurs earlier in the delivery route and has space left for additional bikes. Specifically, we distribute the $Q_i - x_i^{NH}$ excess bikes to stations $i'(i' < i)$ that are visited before i and are such that $x_i^{NH} < Q_i$. We provide more details regarding how this is done in Appendix 2.c.

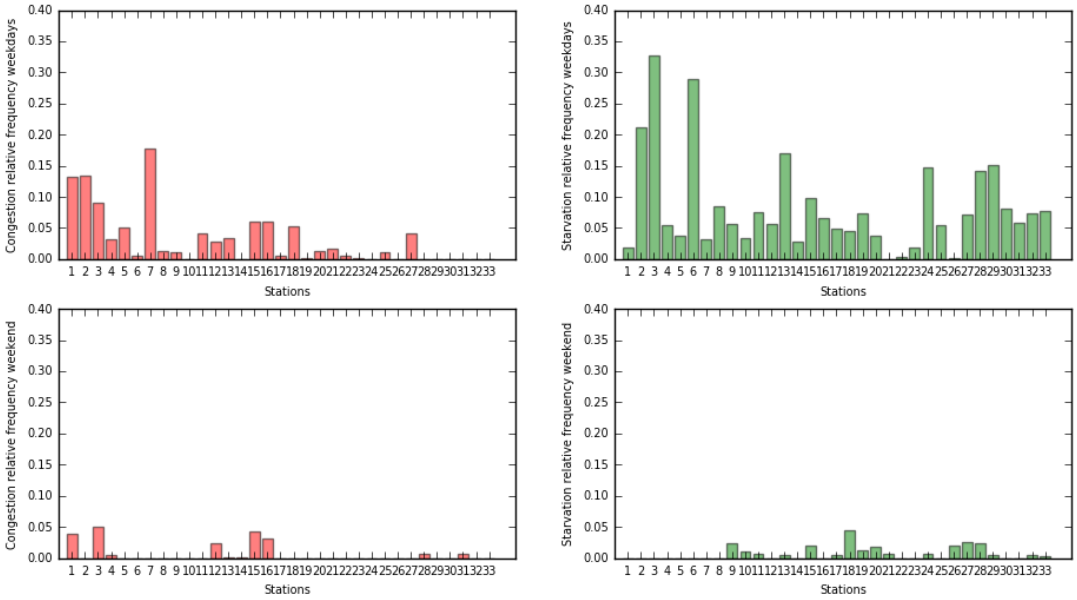
3.5 Numerical Results

In this section, we present and analyze the results of our computational study. With this study, we seek to assess the benefits of modeling uncertainty. We also compare the quality of the initial allocation plan prescribed by the solution to the stochastic program with an estimate based on historical data of how bikes were allocated in practice. Finally, we benchmark the performance of the Newsvendor-based heuristics against the solutions provided by the stochastic program.

Our computational study is based on the bike sharing system in San Francisco, CA. Specifically, we use publicly available data (open data) to simulate the use of different initial allocation plans for that system, and calculate performance metrics related to customer service and cost. As such, we first illustrate the state of art of the San Francisco system which also represents the motivation for our work. After that, we describe how we generated an instance of the stochastic program based upon this system, as well as how we simulated the use of those initial allocation plans in the context of that system. We then present the results of our analysis. All computational experiments were run on a computer with 8 GB of RAM and a 2.70 GHz CPU. All software was implemented in Python 3.6.1, with optimization problems solved with Gurobi 7.5.1.

3.5.1 State of art of San Francisco bikesharing system

To estimate the magnitude of congestion and starvation in a realistic setting, we simulated the bike sharing system of the city of San Francisco. This simulation was based on publicly available data regarding ridership and bike availabilities in the time interval of 6 am to 11:59 am for the months of May through August, 2016. We illustrate in Figure 3.2a the percentage of minutes with congestion (computed with respect to the overall considered time interval), by station, and with separate graphs for weekdays and weekends. We see that congestion is more likely during the week, with some stations seeing an event 20% of the time. Figure 3.2b is similar, only it reports the percentage of minutes with starvation. Here we see an even greater frequency, with some stations experiencing a starvation event in over 30% of the simulated overall time interval.



(a) Congestion frequency during weekdays and weekend

(b) Starvation frequency during weekdays and weekend

Figure 3.2: Congestion and starvation average frequencies in the real case for year 2016, considering the months from May to August and the time interval between 6 a.m. and 11.59 a.m.

3.5.2 Test setting

Figure 3.3 illustrates the bike sharing system of the city of San Francisco, with markers indicating its 33 stations (in August, 2016). Ridership data for this system can be found at the website www.bayareabikeshare.com/open-data. We derived our instances, and simulation model, from ridership data from the summer months (May through August) in the period of time between August, 2013 (the launch of the bike sharing service) and August 2016. We note that when we first accessed the website, it was indicated that the system consisted of 350 bikes, and we used that number in both settings (instances and simulation model).

In our computational study, both the allocation penalty, f_i, and rebalancing penalty, $t_{i,i+1}$ are set to the same value for each station. Specifically, we set $f_i = 1$, and $t_{i,i+1} = 2$ for all stations i. We penalize congestion and starvation by modeling that when a user can not return (withdraw) a bike to (from) the station they desire, they will instead walk to the next-closest station. As such, we set both $c_i = p_i = \kappa(1 + \min_{j \in \mathcal{I}: j \neq i} \delta_{ij})$, for all stations i, where δ_{ij} is the distance between station i and j calculated as the geodesic distance using the great-circle distance formula (Banerjee [7] (2005)).

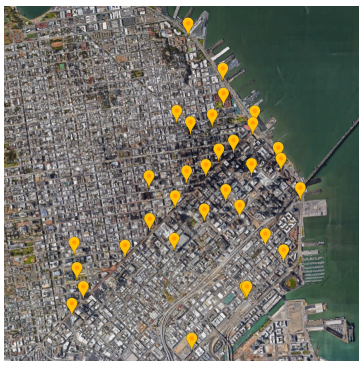


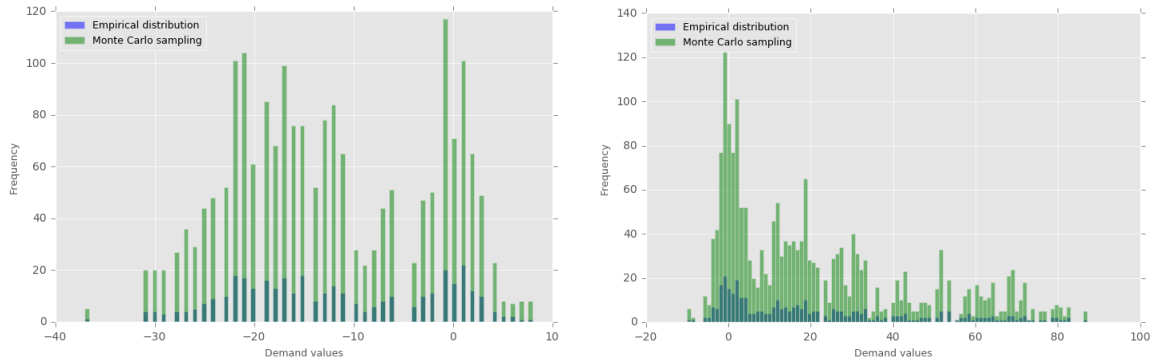
Figure 3.3: San Francisco bike sharing system

Specifically, we set $\kappa = 46$, so that, in most of cases, the model redistributes bikes from a station when there is extra inventory to one of the following stations (since it is cheaper to rebalance than having congestion or starvation). Regarding other model parameter values, we presume the vehicle used for rebalancing bikes has a capacity of 25 bikes. This is the same value that was used in Forma, Raviv, and Tzur [35] (2015).

Scenario generation

Scenario generation is an important part of the modelling process, since a bad scenario tree can lead to a not meaningful solution of the optimization problem. We recall that a typical assumption of stochastic programming is that the distribution of the random variable is known. However, in most practical applications, the distributions of the stochastic parameters have to be approximated by discrete distributions with a limited number of outcomes. The discretization is called a scenario tree. We assume that the random variable given by the demand at each station, has a finite number of possible outcomes at the end of the considered period, assumed to be exogenous to the problem. Consequently, the probability distribution is not influenced as well by decisions. Making these assumptions, we can represent the stochastic process demand d_i^s, $i \in \mathcal{I}$, $s \in \mathcal{S}$, using a scenario tree which contains a root and a finite set of leaves. In the problem under consideration, the random vector is high-dimensional, and presents complicated dependencies among stations. These factors make the uncertainty very difficult to represent. For this reason, we derived an empirical distribution of each station's demand as inverse of the Kaplan-Meier estimate of the cumulative distribution function (also known as the empirical cdf) of the real historical ridership data that we collected and that we denote as d_i^n, $i \in \mathcal{I} \setminus \{I\}, n = 1, \ldots, N$, where N is the total number of collected data. Specifically, for each day and each station, we computed the number of withdrawn and returned bikes between 6 am and 11:59 am, and set the demand for that

day as the difference between those two numbers (withdrawn-returned). Since we add penalties in the objective function for extra and excess inventory and for stockout, we do not bound the demand values to stations capacities. From the empirical demand distributions, several scenario trees of increasing size are then generated according to a Monte Carlo sampling procedure. Pseudo-code (**??**) presents the details for scenario generation. Finally, figures 3.4a and 3.4b illustrate for two chosen stations in San Francisco a comparison of the empirical distribution based on 369 real observations and Monte Carlo sampling based on 2,000 observations. Results show that the two patterns follow a similar behaviour with only a difference in scale, due to the number of considered samples.



(a) Station 1 **(b) Station 2**

Figure 3.4: Empirical distributions and Monte Carlo sampling for two different stations

Determining the size of the scenario tree

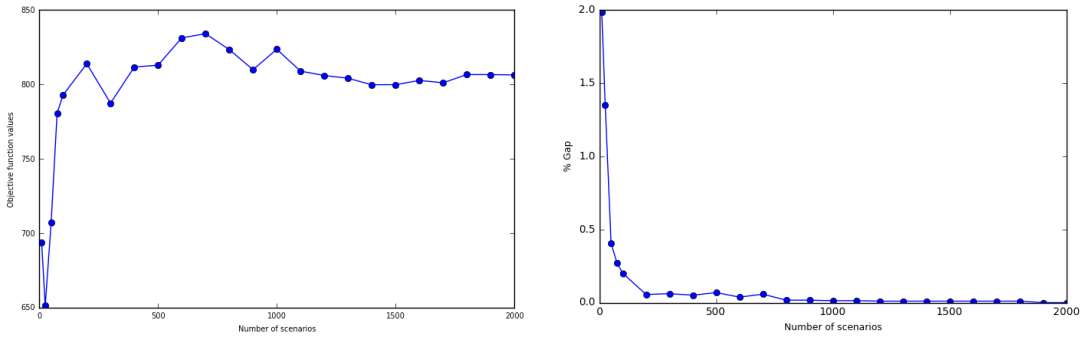
In order to understand how many scenarios are needed to obtain a stable objective function, we performed both an in-sample and out-of-sample stability analysis of our stochastic program, following the procedure described in Kaut and Wallace [51] (2007). We illustrate the results of these analyses in Figures 3.5a and 3.5b. Regarding the in-sample analysis, we solved the stochastic program for scenario trees of increasing size. Figure 3.5a indicates that the objective function value stabilizes with 1,200 scenarios. However, we have to remember that in-sample values are not directly comparable. To be able to estimate the effect of using a larger scenario tree, we have to compare the out-of-sample costs. For this purpose, we declare a scenario tree with 2,000 scenarios to be the true representation of the real world, and we use it as a benchmark to evaluate the cost of the optimal solutions obtained using scenario trees with a smaller size. We see in Figure 3.5b that convergence is nearly monotonic and decreasing with a percentage gap under 0.1%. We notice that, even if the model is an integer linear program, an optimal integer solution has always been obtained by solving the continuous relaxation

Pseudo-code 1: Scenario generation process

Input: $d_i^n \in \mathbb{Z}$, $i \in \mathcal{I} \setminus \{I\}$, $n = 1, \ldots, N$
for $i \in \mathcal{I} \setminus \{I\}$ **do**
  $\mathcal{K} := \{n', n'' = 1, \ldots, N : d_i^n \neq d_i^n\}$
  if $d_i^n < d_i^n$ **then**
    $n' < n''$
  end
  for $k \in \mathcal{K}$ **do**
    $cdf_i[k] := \frac{\sum \mathbb{1}}{N}$
    $inv.cdf_i[k] := d_i^k$
  end
  for $s \in \mathcal{S}$ **do**
    sample a random number ("*random*") in [0,1]
    for $k \in \mathcal{K}$ **do**
      if $random <= cdf_i[k]$ *and* $random > cdf_i[k-1]$ **then**
        $d_i^s = inv.cdf_i[k]$
      end
    end
    return d_i^s
  end
end

at the root node. As a consequence, considering that the continuous relaxation of the model with 1,200 scenarios required 66 seconds to solve, whereas one with 2,000 required 115 seconds, we base our computational study on a set of 1,200 scenarios. We note that we presume the scenarios are equi-probable, i.e. we set $pr^s = \frac{1}{|\mathcal{S}|}, \ \forall s \in \mathcal{S}$.

Notice that, as highlighted in Raviv and Kolka [76] (2013), computing demand as the difference between the number of withdrawn and returned bikes during a time interval yields what could be thought of as a "steady-state" demand and ignores the dynamics of the system. As an example, consider a station where first a bike is withdrawn and then one is returned, and no other bikes are withdrawn or returned. We would compute the resulting station demand as 0, which in turn would indicate to the model that no bikes need to be allocated to that station (unless they were allocated to that station only to be rebalanced later). With zero bikes allocated, however, the first withdrawal request can not be satisfied.



(a) In-sample stability analysis results **(b) Out-of-sample stability analysis results**

Figure 3.5: In sample and out-of-sample analysis

As such, we discuss in Appendix 2.b our method for estimating the number of bikes withdrawn from a station before any are returned, from which we derive the parameter values $\underline{x}_i$.

Simulation of the inventory levels

We also use historical ridership data to simulate the performance of our initial allocation of bikes to stations during the week of June 20, 2016 to June 26, 2016. Given a day during this week, we simulate the movement of bikes based on rides taken on that day, and record statistics related to congestion and starvation. Specifically, regarding congestion, we record the number of times during the simulation a user wants to return a bike to a station, but it is full. Similarly, regarding starvation, we record the number of times during the simulation a user wants to withdraw a bike from a station, but it is empty. Then, we have a final inventory level at each station, $I_i^{final} = \overline{I}_{i0} + x_i - w_i + s_i$, where w_i and s_i stand for the number of withdrawn and returned bikes, respectively, at station i, defined in the interval of integer numbers $[0, Q_i]$, from which we solve the second stage of our stochastic program in order to determine how rebalancing should be done and the number of bikes rebalanced. Similarly, we calculate the total number of rebalanced bike miles, as $\sum_{i,i+1} \delta_{i,i+1}\bar{y}_{i,i+1}$, where $\bar{y}_{i,i+1}$ indicates how many bikes should be rebalanced from station i to station $i+1$ according to the rebalancing plan.
Since it could be interesting to observe how the system performs after rebalancing is done, an alternative perspective on these final inventory levels is to calculate a fill rate-type statistic, wherein we estimate the percentage of future bike withdrawals they can satisfy. We base this estimate on the same set of scenarios described above. For a given station, i, and scenario s, we compute the fill rate as follows:

$$FR_i^s = \begin{cases} 1 & \text{if } I_i^{final} - d_i^s \geq 0 \\ \frac{I}{d} & \text{otherwise} \end{cases} \tag{3.25}$$

We then compute an overall fill rate by averaging this statistic over all stations and scenarios.

3.5.3 Analyzing the value of uncertainty and the quality of the expected value solution

To assess the value of modeling uncertainty, we compute the *Value of the Stochastic Solution (VSS)* (Birge and Louveaux [12] (2011), Kall and Wallace [50] (1994), Maggioni, Allevi, and Bertocchi [58] (2016)). To do so, we solve the stochastic program presented above on the benchmark scenario tree, to get its optimal objective function value, *RP*. We then solve the *Expected Value Problem (EV)*, which is obtained by solving a deterministic variant of our stochastic program, in which the random demand parameters are replaced with their expected values, rounded to the nearest integer (bikes demands cannot be represented by fractional values). We then evaluate how the deterministic solution performs in the stochastic setting by computing the *Expectation of Expected Value*, *EEV*, obtained by fixing the first-stage expected value decisions in the stochastic program and we compute the (Relative) *Value of Stochastic Solution*

$$VSS = (EEV - RP)/RP = 41.15\%,$$

suggesting that significant gains can be realized by solving the stochastic program versus the expected value approach.

We also assess how well the initial allocation plans from both the EV problem and the stochastic program perform in our simulation model, and report statistics related to the performance of each plan in Table 3.1. We see that while the stochastic program allocates 40% more bikes, it leads to a much smaller frequency of starvation and higher fill rate. The initial plan prescribed by the stochastic program also requires less rebalancing. However, as it allocates more bikes, the plan prescribed by the stochastic solution also yields a higher frequency of congestion.

We next seek to understand why the solution to the EV problem performs poorly in comparison to the solution to the stochastic program. To do so, we compute two more indicators: the *Loss of Using the Skeleton Solution (LUSS)* and the *Loss of Upgrading the Deterministic Solution (LUDS)* defined in Maggioni and Wallace [56] (2012).

To compute the $LUSS$, we examine the solution to the EV problem to determine the subset of stations $\bar{\mathcal{I}}$ to which it allocates more bikes than their initial requirement $\underline{x}_i$. We then solve the stochastic program, fixing $x_i = \underline{x}_i$ for stations $i \in \mathcal{I} \setminus \bar{\mathcal{I}}$. We refer to the objective function value of the optimal solution to this problem as the *Expected Skeleton Solution Value (ESSV)* and found the (Relative) $LUSS$ measure,

$$LUSS = (ESSV - RP)/RP = 7.95\%.$$

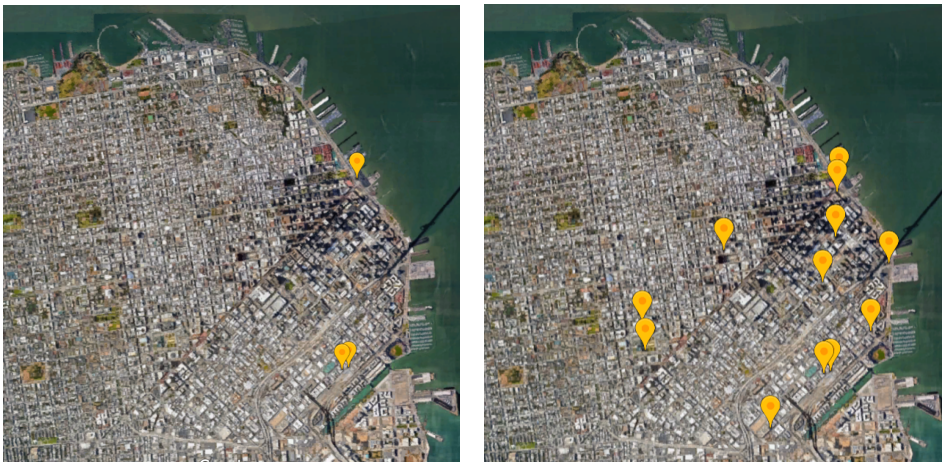
	SP					EEV				
	% Congestion	% Starvation	Miles * Rebalanced Qty	Expected Fill Rate	Extra Inventory Qty	% Congestion	% Starvation	Miles * Rebalanced Qty	Expected Fill Rate	Extra Inventory Qty
Mon	8.81%	31.30%	19.95	85.86%	149	5.56%	39.57%	50.65	86.28%	151
Tue	9.21%	28.74%	22.17	86.28%	149	6.53%	38.32%	49.70	86.45%	153
Wed	7.27%	31.11%	16.85	87.55%	158	4.44%	40.50%	46.55	88.13%	163
Thu	6.94%	27.88%	15.34	86.43%	141	4.49%	36.01%	42.78	86.17%	145
Fri	6.68%	25.98%	19.89	86.42%	105	3.23%	36.55%	51.59	87.03%	109
Sat	7.62%	21.36%	3.36	88.22%	37	0.00%	23.30%	3.36	85.26%	37
Sun	0.00%	20.83%	2.31	91.32%	19	0.00%	25%	2.31	88.04%	19
Average	**6.65%**	**26.74%**	**14.27**	**87.44%**	**108.29**	**3.46%**	**34.18%**	**35.28**	**86.77%**	**111**
Tot delivered bikes	154					110				

Table 3.1: Simulation-based comparison of solutions to stochastic and deterministic problems.

The positive $LUSS$ value means that the expected value solution selects the wrong quantities to deliver to the wrong stations and its structure (skeleton) cannot be inherited in a stochastic environment. Relatively, we illustrate in Figure 3.6a the stations in $\bar{\mathcal{I}}$ in the solution to the EV problem and in Figure 3.6b the analogously-determined stations in the solution to the stochastic program SP. From the figures we can conclude that the solution of the EV model allocates bikes to too few stations compared to the SP one. Specifically, in the EV solution only 3 stations receive a higher number of bikes than the initial requirement, compared to the SP in which these stations are 12. We also note that the 3 stations activated in the EV solution are the same activated in the SP solution.

This result could justify a deeper investigation of the quality of the Expected Value Solution by computing the *Loss of Reduced Costs-based Variable Fixing* ($LRCVF$) as the difference between the optimal values of the stochastic problem and its reduced version, obtained by fixing a certain number of variables taking into account the information from the reduced cost of the expected value solution (for more details see Crainic, Maggioni, Perboli, and Rei [25] (2017)).

With the $LUDS$, we seek to determine whether the solution to the EV problem is upgradeable, i.e. that it can be used as a starting point for generating a high-quality solution to the stochastic program. To do so, we solve the stochastic program, albeit with additional constraints ensuring that the values of the first-stage variables are at least as large as their values in the optimal solution to the EV problem. We refer to the objective function value of the optimal solution to this restricted stochastic program



(a) First-stage *EV* skeleton solution **(b) First-stage *SP* skeleton solution**

Figure 3.6: Distributions of bikes to stations in the EV and SP solutions

as the *Expected Input Value (EIV)* and compute a relative $LUDS$ measure as

$$LUDS = (EIV - RP)/RP = 0\%.$$

The result means that the EV solution is *perfectly upgradable*, indicating that solving the EV problem can be a good start for solving the stochastic program. Besides, we obtained that by first solving the EV problem, and then the LUDS-restricted stochastic program, the total solution time is reduced by 10% of what it is needed to solve the stochastic program from scratch.

Finally, we study the correlation between the variance in demand at a station and the number of bikes allocated to that station in both the solution to the stochastic program and the solution to the EV problem. We present a scatter plot of demand variance against number of allocated bikes in Figure 3.7. We observe a high correlation (0.84) in the solution to the stochastic program, suggesting that the stochastic program allocates more bikes to stations that have a greater variability in station demand. However, we also see a correlation of 0.64 in the solution to the EV problem. While this may seem counter-intuititive, as the EV problem does not recognize variance, there is a high correlation between the average and variance of demand at stations. Thus, we conclude that this correlation coefficient of 0.64 reflects the solution to the EV problem allocating bikes to stations with high average demand.

3.5.4 Newsvendor-based heuristics

We next seek to assess the quality of the initial allocation plans prescribed by the three Newsvendor heuristics previously presented. Recall that they differ with respect to how the critical ratio is cal-

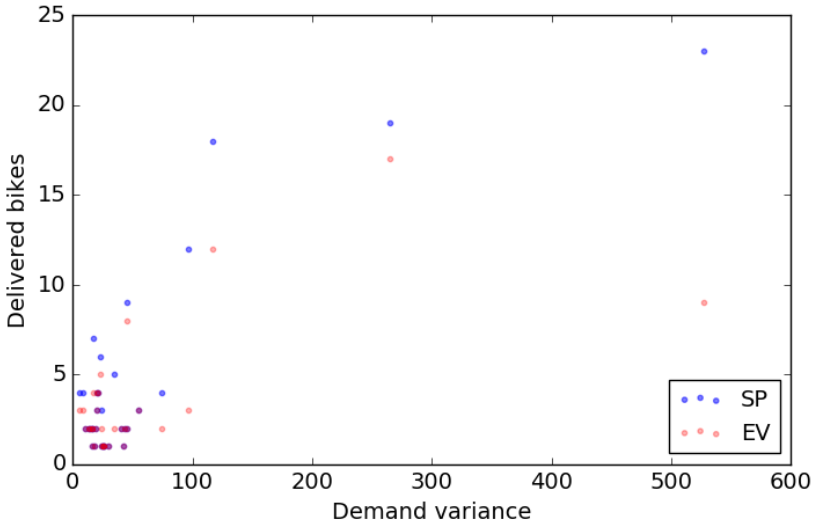


Figure 3.7: Number of allocated bikes vs demand variance.

culated, and thus are labeled CR1, CR2, and CR3. We benchmark the performance of these initial allocation plans against the plan prescribed by the solution to the stochastic program by looking at the objective function value and the computational time (Table 3.2) and at our simulation results (Table 3.3).

First, we notice that the stochastic program chooses to allocate more bikes, which in turn leads to higher first-stage costs with respect to the values obtained through the heuristics. However, the second-stage costs are lower in the stochastic program solution, while the computational times are higher than the heuristics. We also observe that the SBH3 performs quite well, since it leads to a second-stage objective value gap of only 1.76% with respect to the stochastic program, obtained with a considerable reduction in the computational time. Furthermore, we see that, as expected, the higher number of allocated bikes leads to a higher congestion frequency. However, at the same time, the allocation plan results in a much lower starvation frequency and less rebalancing. We note that of the Newsvendor-based heuristics, the third method for computing the critical ratio performs the best. However, CR3 involves more rebalancing, and has a higher frequency of starvation. We attribute that the stochastic program outperforms the Newsvendor-based heuristics to the fact that it recognizes the opportunity to remodel at a later point in the day, the total initial availability and the stations' capacities.

	SP			SBH1			SBH2			SBH3		
	1st stage cost	**2nd stage cost**	**sec**	**1st stage cost**	**2nd stage cost**	**sec**	**1st stage cost**	**2nd stage cost**	**sec**	**1st stage cost**	**2nd stage cost**	**sec**
MON	155	568.35	66	116	594.38	0.04	113	594.38	0.06	145	574.23	0.05
TUE	155	552.62	66	116	573.88	0.04	113	573.88	0.04	145	558.71	0.04
WED	155	546.82	66	116	577.91	0.04	113	579.91	0.03	145	561.41	0.06
THU	155	500.20	66	116	523.45	0.04	113	523.45	0.04	145	508.20	0.04
FRI	155	374.66	66	116	408.22	0.04	113	408.22	0.04	145	389.29	0.05
SAT	155	138.13	66	116	138.13	0.04	113	138.13	0.04	145	138.13	0.03
SUN	155	68.87	66	116	68.87	0.04	113	68.87	0.04	145	68.87	0.06
AVERAGE	155	392.81	66	116	412.12	0.04	113	412.41	0.04	145	399.83	0.05
AVERAGE %	-	-	66	-33.62%	+4.69%	-1649%	-37.17%	+4.75%	-1649%	-6.90%	+1.76%	-1319%

Table 3.2: Objective function and computational time results comparison of the stochastic program and the SBH's.

$\max(x_i^{NW}, \underline{x}_i)$	SP					SBH1					SBH2					SBH3				
	% Congestion	% Starvation	Miles * Rebalanced Qty	Expected Fill Rate	Extra Inventory Qty	% Congestion	% Starvation	Miles * Rebalanced Qty	Expected Fill Rate	Extra Inventory Qty	% Congestion	% Starvation	Miles * Rebalanced Qty	Expected Fill Rate	Extra Inventory Qty	% Congestion	% Starvation	Miles * Rebalanced Qty	Expected Fill Rate	Extra Inventory Qty
Total delivered bikes	155					116					113					145				
MON	8.81%	31.30%	19.95	85.86%	149	5.56%	38.19%	50.47	86.00%	151	5.56%	38.78%	50.47	86.00%	151	7.85%	31.69%	35.31	86.24%	147
TUE	9.21%	28.74%	22.17	86.48%	149	6.53%	35.93%	48.99	85.87%	153	6.53%	36.73%	48.99	85.87%	153	8.25%	29.34%	36.42	86.68%	148
WED	7.47%	31.11%	16.85	87.55%	157	4.44%	38.83%	45.48	87.72%	163	4.44%	39.88%	45.66	87.90%	163	6.06%	32.78%	30.64	87.72%	160
THU	6.94%	27.88%	15.34	86.43%	141	4.49%	34.38%	42.43	85.93%	145	4.49%	36.27%	42.43	85.93%	145	6.12%	28.51%	29.13	86.37%	141
FRI	6.68%	25.98%	19.89	86.42%	104	3.23%	34.48%	50.88	86.70%	109	3.23%	35.63%	50.88	86.70%	109	5.76%	27.59%	33.69	87.02%	107
SAT	7.62%	21.36%	3.36	88.26%	37	0.00%	23.30%	3.36	85.78%	37	0.00%	23.30%	3.36	85.56%	37	1.90%	22.33%	3.36	88.03%	37
SUN	0.00%	20.83%	2.31	91.34%	19	0.00%	25.00%	2.31	88.50%	19	0.00%	25.00%	2.31	88.29%	19	0.00%	20.83%	2.31	90.40%	19
AVERAGE	**6.68%**	**26.74%**	**14.27**	**85.45%**	**108**	**3.45%**	**32.87%**	**34.85**	**86.64%**	**111**	**3.46%**	**33.66%**	**34.87**	**86.61%**	**111**	**5.13%**	**27.58%**	**24.41**	**87.49%**	**108.43**

Table 3.3: Simulation results comparison of the stochastic program and the SBH's.

3.5.5 A comparison with the implemented system

We finish with a comparison of how the plan prescribed by the stochastic program performs relative to what we determined was the initial allocation plan in the actual system. We derived the initial allocation of bikes to stations from station status data by collecting the number of bikes at each station at 6 a.m., for each day of the considered week. We compare the performance of the two initial allocation plans with our simulation model. We present statistics regarding how each plan performed in Table 3.4. We see that the stochastic program allocated far fewer bikes (45% fewer), which in turn lead to less congestion and rebalancing. However, not surprisingly, we saw an increase in the starvation frequency.

	Real system						SP					
	Total bikes at 6 a.m.	% Congestion	% Starvation	Miles * Rebalanced Qty	Expected Fill Rate	Extra Inventory Qty	Total bikes at 6 a.m.	% Congestion	% Starvation	Miles * Rebalanced Qty	Expected Fill Rate	Extra Inventory Qty
Mon	272	17.24%	25.39%	40.09	90.92%	113	155	8.81%	31.30%	19.95	85.86%	149
Tue	275	14.97%	19.76%	38.19	88.58%	109	155	9.21%	28.74%	22.17	86.28%	149
Wed	281	15.76%	23.38%	24.40	91.49%	118	155	7.47%	31.11%	16.85	87.55%	157
Thu	289	13.88%	22.22%	38.76	90.92%	101	155	6.94%	27.88%	15.34	86.43%	141
Fri	285	10.14%	14.94%	22.73	91.46%	85	155	6.68%	25.98%	19.89	86.42%	104
Sat	293	4.76%	2.91%	3.36	94.03%	31	155	7.62%	21.36%	3.36	88.26%	37
Sun	288	17.39%	14.58%	2.12	92.80%	18	155	0.00%	20.83%	2.31	91.34%	19
Average	**283.29**	**13.45%**	**17.58%**	**24.24**	**91.46%**	**82.14**	**155**	**6.68%**	**26.74%**	**14.27**	**85.45%**	**108**

Table 3.4: Comparison of implemented plan and plan from stochastic program

Next, to normalize our comparison, we add a constraint to the stochastic program to ensure that it allocates the same total number of bikes on each day as used in the real system. We present the statistics related to that plan in Table 3.5. Here, we see that the allocation plans prescribed by the stochastic program outperform the actual allocation on each day and in each category. We view these results as a strong indicator of the impact the stochastic program we propose could have in practice.

3.6 Managerial insights

In this section, we present some managerial insights which can be valuable for practitioners. From the literature and from the case study of the bikesharing system of San Francisco (see subsection 3.5.1), one of the problems in common to all bikesharing systems is that stations are full or empty very often. For this reason, we believe that our approach could be extended to other bikesharing systems and, without loss of generality, the following managerial insights could be valid.

First, managers should apply a stochastic model since demand is uncertain. By applying a deterministic

	Real system						SP when initial number of bikes equal to the real system					
	Total bikes at 6 a.m.	% Congestion	% Starvation	Miles * Rebalanced Qty	Expected Fill Rate	Extra Inventory Qty	Total bikes at 6 a.m.	% Congestion	% Starvation	Miles * Rebalanced Qty	Expected Fill Rate	Extra Inventory Qty
Mon	272	17.24%	25.39%	40.09	90.92%	113	272	12.07%	18.70%	28.21	91.92%	127
Tue	275	14.97%	19.76%	38.19	88.58%	109	275	11.90%	18.56%	37.73	92.09%	117
Wed	281	15.76%	23.38%	24.40	91.49%	118	281	9.70%	22.97%	24.40	93.46%	140
Thu	289	13.88%	22.22%	38.76	90.92%	101	289	9.18%	16.98%	24.43	93.64%	114
Fri	285	10.14%	14.94%	22.73	91.46%	85	285	7.83%	13.79%	20.20	93.54%	93
Sat	293	4.76%	2.91%	3.36	94.03%	31	293	7.62%	2.91%	4.16	96.90%	33
Sun	288	17.39%	14.58%	2.12	92.80%	18	288	4.35%	8.33%	2.31	97.80%	19
Average		**13.45%**	**17.58%**	**24.24**	**91.46%**	**82.14**		**8.95%**	**14.61%**	**20.21**	**94.19%**	**91.86**

Table 3.5: Comparison of implemented plan and plan from stochastic program, when allocating same number of bikes.

model, the solution is very sub-optimal, since it delivers too few bikes to too few stations, leading to a higher frequency of starvation, more rebalancing and inventory and a lower demand fill rate. However, practitioners could reduce the computational effort to obtain the solution of the stochastic program by upgrading the deterministic solution. The heuristic procedure we proposed can be valuable, especially if this problem must be solved multiple times in a day. Second, the number of allocated bikes represents an indicator of how big the fleet size should be. According to this, managers should carefully evaluate whether to allocate the total number of available bikes at the depot. In our case study, only a fraction of the total availability of bikes should be delivered in order to reduce the rebalancing and the risks of congestion and those deriving from the allocation. Third, rebalancing is fundamental in order to adjust the allocation decision over time. As a matter of fact, this strategy is able to introduce more flexibility to better manage a bikesharing system. Finally, managers should carefully select the stations towards which bikes should be delivered, as each station is characterized by different features, such as the number of requests for bikes and free docks, capacity and distance to the closest station.

In our work, we captured all these elements and through the comparison of the allocation decisions suggested by our stochastic program and those actually implemented, we observed that our solution improves the allocation of bikes between stations, as it leads to lower frequencies of congestion and starvation, lower rebalancing and a higher expected fill rate. Finally, we noticed that our stochastic program always outperforms the Newsvendor-based heuristics. Nevertheless, we also noticed that one of them provides a good solution, with a very reduced computational effort.

3.7 Conclusions and future works

In this paper, we studied the problem of determining an initial allocation of bikes to stations, as well as the opportunity to perform re-balancing at a later point in time, in the context of a bike sharing system. One of the challenges in determining this allocation is that there are multiple dimensions along which the performance of such an allocation plan can be measured, with some measuring costs incurred while operating the system and others measuring the quality of the service experienced by users of the system. As a result, we present a two-stage stochastic program wherein the first-stage variables determine this initial allocation of bikes, and the second-stage variables determine how bikes are rebalanced at a point later in the day. Another challenge in this setting is determining how to measure demand, as, like a rental system, bikes are both withdrawn and returned from individual stations.

We performed a computational study based upon historical ridership data from the bike sharing system of the city of San Francisco. In particular, we used this data to derive a simulation model wherein we can estimate the performance of an initial allocation plan along multiple dimensions. With that study, we first established that by not recognizing variability in bike station demand, the deterministic problem allocated too few bikes to too few stations. However, we also established that the time required to produce a high-quality solution to the stochastic program can be reduced by first solving its deterministic counterpart. Through the comparison with the Newsvendor-based heuristics, we noticed that, concerning the indicators used to evaluate the simulation, the initial allocation plan prescribed by the stochastic program outperformed those produced by the Newsvendor-based heuristics. However, through the heuristics, we are able to solve the problem with less computational effort with respect to the stochastic program and the objective function gap is small, especially for SBH3. We also compared the performance of the initial allocation plan prescribed by the stochastic program with what we estimated was the initial allocation plan for a given week of historical data. We saw that the stochastic program produced a much better initial allocation of bikes than what we estimated was done in practice. Finally, we proposed managerial insights which could be valuable for practitioners in order to manage a general bikesharing system.

Regarding future work, we believe the next logical step in this research is to consider a multi-stage stochastic optimization model that recognizes that rebalancing can occur multiple times throughout the day. This variant will be compared with the two-stage formulation provided in this paper, by means of rolling horizons approaches (see Bertazzi and Maggioni [11] (2018)). Finally, another extension is to model that multiple vehicles can be used to support rebalancing and the possibility that their routes can change from one day to the next.

Acknowledgments

The authors thank the Transportation and Logistics Society (TSL) as this research has been supported in part by the TSL Cross Region Doctoral Grant.

Chapter 4

Workforce production planning under uncertain learning rates

Authors: Rossana Cavagnini[1], Mike Hewitt[2] and Francesca Maggioni[3]

(This chapter is under evaluation in International Journal of Production Economics. Manuscript Reference Number: IJPE-D-18-01597)

Keywords: Production planning, Stochastic programming, Stochastic learning rates, Cross-training

[1]University of Bergamo, Via dei Caniana, 2, Bergamo, Italy, e-mail: r.cavagnini@studenti.unibg.it
[2]Loyola University Chicago, Chicago, U.S.A, e-mail: mhewitt3@luc.edu
[3]University of Bergamo, Via dei Caniana, 2, Bergamo, Italy, e-mail: francesca.maggioni@unibg.it

4.1 Introduction

For a product manufacturer, meeting customer demands in a cost-effective manner requires both capacity and efficient use of that capacity. This is particularly true when capacity is scarce, which is often the case when products are new and production capacity is partially dictated by human labor. Research (Jin, Hewitt, and Thomas [49] (2018), Valeva, Hewitt, Thomas, and Brown [88] (2016)) has shown that if an organization recognizes the individual learning process, it can enhance the development of its capacity with its production planning decisions. Specifically, an organization can increase its overall capacity and reduce its costs with an understanding that the work that the employees do today will increase their productivity in the future. However, these studies, like much of the research on human learning in the context of production planning, assume that the rate of this increase in productivity (often referred to as the *learning rate*) is known with certainty, which is rarely the case in practice.

However, by examining the performance of past workers, it is possible for an organization to develop a distribution and estimate the corresponding distribution parameters values for the learning rate of an individual worker. As such, in this paper, we study a problem wherein an organization seeks to assign workers to produce multiple types of products to meet known customer demands while recognizing both that learning occurs and that there is uncertainty in its rate. We formulate this problem as a two-stage stochastic program (see Birge and Louveaux [12] (2011), Kall and Wallace [50] (1994)), wherein scheduling decisions are made in the first stage, and production quantity decisions, which are constrained by realized learning rates, are made in the second. One of the challenges associated with embedding recognition of human learning in a mathematical program is that learning models are typically non-linear, including the one we consider in this paper. The computational challenge associated with solving the resulting non-linear program is magnified by the fact that the uncertain learning parameter affects all the possible realizations of the scenarios of our model, and, consequently, the non-linear functions would appear in each learning rate scenario. In this work, we adopt a reformulation technique (Hewitt, Chacosky, Grasman, and Thomas [44] (2015)) that enables the formulation of our stochastic program as a mixed integer linear program that can be solved efficiently, even for instances based on a large number of scenarios.

The solutions of our model are used to derive insights into the tactics product manufacturers should employ to leverage the benefits of human learning without having full knowledge of the learning rates. Specifically, we present and statistically analyze results from an extensive computational study based on solving a large set of instances. To ensure that the insights we derive are generally true, and not an artifact of the instances we consider, we generated the instance set in a statistically-rigorous manner, which we will describe later in detail.

Fundamentally, we believe this paper makes the following contributions. First, we present a stochastic program that organizations can use to effectively plan their production operations by leveraging the benefits of human learning, even when learning rates are not known with certainty. Second, we illustrate how the reformulation technique of Hewitt, Chacosky, Grasman, and Thomas [44] (2015), which was presented for a deterministic model, can be used in a stochastic setting to yield a model that can still be solved in a reasonable run-time. Third, we show the value in recognizing uncertainty in learning rates as well as why not doing so leads to worse plans. Fourth, we present tactics an organization can employ to leverage the benefits of human learning, even when learning rates are not known with certainty.

The remainder of this paper is organized as follows. Section 4.2 presents a review of the literature most closely related to this paper. Section 4.3 describes the production setting we consider in detail, as well as the tactics we seek to assess. In Section 4.4, we present our methodology for assessing those tactics and Section 4.5 presents the experimental setting. Section 4.6 then discusses the results. Finally, Section 4.7 presents conclusions and ideas for future work.

4.2 Literature Review

The research we present in this paper studies two issues that have received attention in the literature on production planning. First, we study how learning, and uncertainty in worker learning rates, should impact worker assignment decisions. Second, we also study how learning, and uncertainty in worker learning rates, can be mitigated by decisions related to cross-training and practice. Consequently, we review the literature that focuses on these two issues.

4.2.1 Learning and worker assignment

While our paper considers how uncertainty in worker learning rates should influence the choice of worker assignments, the first group of works we analyze studies the effects of uncertainty in learning on optimal production levels. These works also study when results from deterministic settings extend to the stochastic setting.

Specifically, in the deterministic setting, two results have been proven that have also been examined in a stochastic setting: (1) that optimal production levels can exceed myopic production levels and (2) recognizing that learning can reduce per-unit production costs can yield higher profit-maximizing production levels, even when price is inversely related to production quantities. Mazzola and McCardle [61] (1996) consider uncertainty in the relationship between cost and cumulative production. They develop a model in which a firm supposes a prior probability distribution of the cost function, and,

as costs are observed over time, its prior belief is updated in a Bayesian manner. They show the existence of the conditions under which the deterministic result (1) (i.e. that optimal production levels exceed myopic production ones) still holds for the stochastic setting. However, they show that the deterministic result (2) does not hold in the Bayesian setting. Stochastic learning curve models are also considered in Mazzola and McCardle [62] (1997), who show that if the learning rate is unknown and can take negative values, the deterministic result (1) does not necessarily hold, since the possibility of having a negative learning rate (intended as a form of forgetting) increases the risk of getting involved in larger production cycles in the myopic case.

Similar to our work, Nembhard and Bentefouet [69] (2012) study the choice of workers' assignments to tasks and prove that, under some specific conditions, the optimal schedule is a schedule with specialized workers, even when learning rates are stochastic. However, they presume demand can be partially met, while we require the demand to be completely met, albeit potentially via outsourcing. Moreover, the objective of their model is to maximize output, while ours is to minimize expected costs. One implication of these different objectives is that our model recognizes value in practicing whereas theirs does not.

Most of the quantitative models of human learning are non-linear (see Dar-El [26] (2013), Anzanello and Fogliatto [4] (2011) and Jaber [48] (2006) for a complete review). In order to embed these models in mixed integer linear programs, that can be solved much more quickly than non-linear programs, Hewitt, Chacosky, Grasman, and Thomas [44] (2015) propose a reformulation technique that models a non-linear learning curve with binary variables and linear constraints. We adapt this reformulation technique to a production scheduling problem where worker learning rates are stochastic.

4.2.2 Cross-training and practicing

Cross training and practicing are two different methods for fostering workforce flexibility, with benefits that have been studied in the literature. Through cross-training, workers increase productivity on multiple tasks, so that they can be re-assigned to different tasks to respond to changes in demand. Through practicing, workers increase their productivity without generating inventory. We first review the papers focusing only on cross-training and then those that also study practice.

A thorough review of the cross-training literature is presented in Qin, Nembhard, and Barnes II [75] (2015). The cross-training literature can be partitioned into two groups: papers that consider learning and those that do not. Belonging to the first group, Olivella, Corominas, and Pastor [70] (2013) present a model to assign tasks to workers, which, differently from our paper, considers cross-training targets, and, as in our work, tasks with due dates. However, they do not draw any conclusions nor managerial insights on the type of relationship between cross-training and workers' characteristics.

Differently from our paper, in which we assume that workers have no initial experience in any task, some papers investigate how to allocate already cross-trained workers which learn. Corominas, Olivella, and Pastor [24] (2010) present a model to allocate cross-trained workers to tasks (considering that the experience on a task is correlated to the experience at another task) with the objective of minimizing the completion time. Sayın and Karabatı [83] (2007) propose a model whose objective is assigning cross-trained workers across departments in order to maximize both the department utility and the improvement in workers' skills. They find out that incorporating the skill improvement function in the model leads to a more effective worker assignment, even if a loss in the department utility can result due to a loss of efficiency. Heimerl and Kolisch [39] (2010) develop a model to minimize the costs of performing an amount of project work in which the decision is to assign tasks to cross-trained workers considering learning, forgetting and the company's desired skill composition as constraints. In this context, the whole company's knowledge is enriched if faster learners specialize. McCreery and Krajewski [64] (1999) and McCreery, Krajewski, Leong, and Ward [65] (2004) evaluate the performance of different cross-training configurations and the general result is that the advantage of workforce flexibility depends on the characteristics of the operating environment. Marentette, Johnson, and Mills [60] (2009) propose a procedure for evaluating job pairings which could be assigned to the same worker by comparing the cost for cross-training to the benefits of increased staffing efficiencies.

Among the literature assuming a fixed individual productivity (i.e. learning is not recognized), Campbell [17] (1999) introduces a model to allocate cross-trained workers to a multidepartment setting. The author shows that even if workers are little cross-trained, a company can strongly benefit from cross utilization, and beyond a certain value, the contribution of additional cross-training decreases. Brusco and Johns [15] (1998) evaluate different configurations with different cross-trained workers levels in which the objective is to minimize workforce staffing costs, by minimizing the minimum labor requirements. They prove that asymmetric cross-training structures which allow chaining of employee skill classes across work activity categories are strongly beneficial. Slomp and Molleman [85] (2002) analyze four different cross-training configurations through a simulation model with special considerations for some indicators evaluating the effectiveness and efficiency of a team. Their results show that, even if a cross-training policy performs well, a fully cross-trained workforce is not always beneficial. In this group, another research stream addresses the problem of finding the optimal mix of specialized versus cross-trained workers. Agnihothri, Mishra, and Simmons [2] (2003) use a simulation model in order to establish the right number of workers which should be cross-trained and those who have to specialize minimizing the cost for services and customer delay. Moreover, they provide different recommendations by investigating the impact of different service parameters (such as the number of workers, their utilization, the coefficient of variation of service time) on the optimal workforce mix (for example, they

find out that with a higher number of workers, cross-trained workers should decrease). Agnihothri and Mishra [1] (2004) propose a simulation model for comparing different cross-training configurations for a system with three task types where each worker has a primary skill and the management can decide to cross-train workers so that secondary skills can be aquired. They draw up insights about the number of workers which should be cross-trained, the number of secondary skills a worker should possess and the efficiency in secondary skills.

Among the papers investigating both cross-training and practicing, Valeva, Hewitt, Thomas, and Brown [88] (2016) develop a task assignment and production planning model (reformulated as highlighted in Hewitt, Chacosky, Grasman, and Thomas [44] (2015)), considering a later-term stochastic demand. They show that, as demand uncertainty increases cross-training increases, that an increase in inventory costs leads to an increase in practice and, that less experienced workers should practice more.

4.3 Production setting and managerial tactics

In this section, we first describe the production setting we consider. We then review the managerial tactics we seek to assess in this paper.

4.3.1 Production setting

We consider a company that seeks to bring multiple new products to market in a cost-effective manner. These products differ in their selling prices and, relatedly, inventory costs. We presume that the company has already committed to supplying these products to customers at regular periods of time, such as a product manufacturer supplying a retail store. Specifically, we presume that at the end of a fixed span of time (e.g. at the end of a week), the company knows how much of each product it must supply. For each unit of demand that can not be fulfilled, the company must pay a cost. This cost could be due to contractual obligations, purchasing the product from another supplier, or some other measure of lost sales.

Items of different products are made in different production cells, with each item converted from raw materials to a finished good by a single worker in each time period. The company employs a workforce that is fixed in size and heterogenous with respect to the impact of experience on an individual worker's productivity. As the products are new, each individual has no prior experience at manufacturing these items. We presume that the company knows with certainty what the steady-state productivity rate will be for each individual on each product. These steady-state productivity rates can differ both by individual and by product. Unlike these steady-state rates, we presume the company only has estimates

of probability distributions for the rate at which an individual's experience impacts their productivity. We also assume the company has estimates of the means and variances of those distributions.

This company seeks to derive a production schedule over a fixed and finite planning horizon that consists of multiple time spans (weeks) during which demand must be met. Each time span is then divided into time periods (e.g. days) during which production is performed. To ease our discussion, we will speak in terms of satisfying weekly demand over the course of a month via daily production planning decisions. We presume that the company determines its production schedule for the entire month, with the schedule indicating which cell each individual will work in on each day (see figure 4.1 for an illustration of the production setting). Once the month begins, on a daily basis the company determines how much each individual should produce of the product it is assigned to for that day. However, the amount each individual can actually produce depends on their productivity rate, which in turn depends on their experience and learning rate.

Finally, the company pays two types of costs. The first is a daily inventory cost associated with holding one item of each product for one day. The second, as mentioned above, is a cost that is incurred for each unit of demand at the end of a week that can not be met from what the company has in inventory. As each of these costs depend on how much has been produced, which in turn depends on individual stochastic learning rates, they are also uncertain. Finally, we presume the company seeks to minimize the expected total cost.

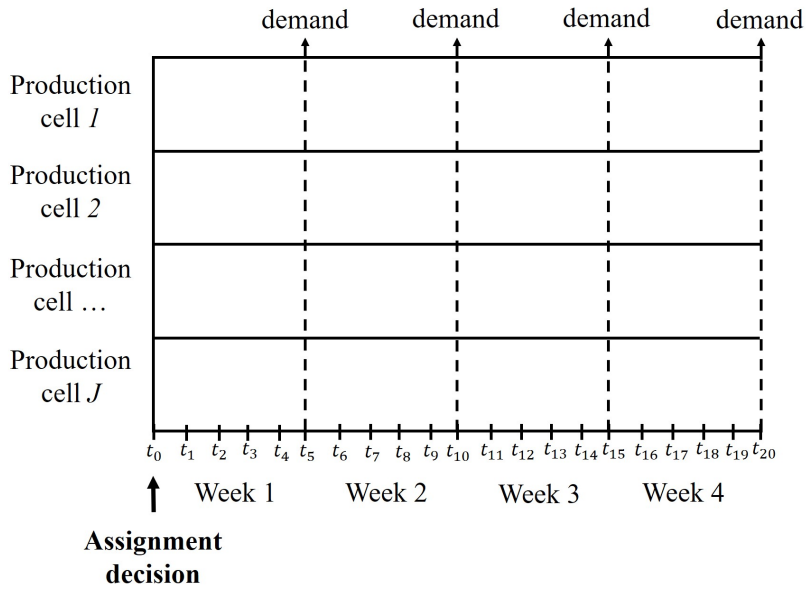


Figure 4.1: Illustration of production setting

4.3.2 Managerial tactics

We next detail the managerial tactics for this production setting. First, we study whether there is value in explicitly recognizing uncertainty in individual learning rates and quantify that value. Then, we study how and why solving the deterministic model may yield plans of greater cost.
After that, we study how uncertainty in individual learning rates should be accounted for when making production planning decisions. Regarding this uncertainty, we presume that the organization knows the distribution type and its parameters, for each individual's learning rates. To compare different workers in terms of their distributions of learning rates, we use the notion of a target capacity for a product (e.g. 5 products/day). We calculate for each worker the probability they will have a productivity rate equal to or greater than that target capacity if they produce that product for a given number of periods. We refer to this probability as a *Target Capacity Probability* (TCP). Similarly, for a given workforce, we determine the average mean learning rate, and then calculate for each worker the probability that their learning rate is less than the workforce average (and hence they learn more quickly). We refer to this probability as the *Faster Than Average Probability* (FTAP).
We present the following hypotheses regarding the tactics an organization should take to minimize their expected costs. The first two hypotheses refer to the frequency with which a worker should produce. The next two hypotheses refer to the number of different products a worker should produce. The last two hypothesis involve which individuals should cross-train and practice, respectively. We next present these hypotheses.

Hypothesis 1. *An organization can minimize its total expected cost by assigning workers with higher TCP to more periods of production.*

Hypothesis 2. *An organization can minimize its total expected cost by assigning workers with lower FTAP to more periods of production.*

Hypothesis 3. *An organization can minimize its total expected cost by assigning workers with smaller variances in learning rates to more products*

Hypothesis 4. *An organization can minimize its total expected cost by assigning workers with higher TCP to more products*

Hypothesis 5. *An organization can minimize its total expected cost by assigning workers with greater variances in learning rates to cross-train less*

Hypothesis 6. *An organization can minimize its total expected cost by assigning workers with lower TCP to practice more*

We will next discuss our methodology for assessing the value in recognizing uncertainty in learning rates when developing production plans, as well as how we test these hypotheses.

4.4 Methodology

Our methodology begins by solving a two-stage stochastic problem. After doing so, and obtaining the resulting optimal solutions, we fit linear regression models to derive relationships between statistics calculated from instance parameter values and statistics calculated from variable values in those solutions. Thus, in this section, we first present the stochastic programming model we solve. We next discuss the statistics we calculate and then the linear regression models we use to derive relationships between these two sets of statistics.

4.4.1 A two-stage stochastic programming model

We consider a multi-product production setting, with a heterogenous workforce of fixed size. As such, we denote the set of workers by $\mathcal{I} = \{1, \ldots, I\}$ and the set of products by $\mathcal{J} = \{1, \ldots, J\}$. Recall that we consider a planning horizon that consists of multiple time spans, wherein at the end of each time span demand should be met. As such, we denote the set of periods in the planning horizon by $\mathcal{T} = \{1, \ldots, T\}$, which is broken up into time spans of length γ. We then denote the first period of the time horizon by t_0 and the subset of periods representing when demand is to be met by $\mathcal{T}'$. As an example, $\mathcal{T}$, can consist of the days in a month, whereas γ consists of the number of days in a week when the company undergoes production, and $\mathcal{T}'$ consists of the end of each week in the month. Specifically, in period $t \in \mathcal{T}'$ the company has agreed to satisfy d_j^t units of demand for product $j \in \mathcal{J}$. However, while not all demand of product $j \in \mathcal{J}$ must be met by internal production, each unit of demand that is not met incurs a cost, which we denote by C_j. Relatedly, for each product $j \in \mathcal{J}$, the company incurs a per-unit, per-period inventory cost, which we denote by h_j.

Our problem presumes that the company determines a production schedule, and then determines the amount to be produced of each product in each day. However, as individual production rates depend on both individual's experience level and learning rate, which is uncertain, these production amounts are random decision variables. As such, we model our problem as occurring in two stages, wherein the production schedule (who works on each product in each period) is determined in the first stage, and the production quantities, which are limited by individual's production rates are determined in the second stage.

As such, we let the binary variable x_{ij}^t indicate whether worker $i \in \mathcal{I}$ is assigned to product $j \in \mathcal{J}$ in period $t \in \mathcal{T}$. We model the impact of worker $i's$ experience level on productivity in product j in

period τ (the current period) with the following exponential learning curve:

$$r_{ij}^{\tau}(\sum_{t=1}^{\tau} x_{ij}^{t}) = I_{ij} + K_{ij}[1 - e^{-\frac{\Sigma}{\cdot}}]. \tag{4.1}$$

This function measures experience in terms of the number of periods during which individual i has produced product j. The function has three parameters: (1) I_{ij}, which indicates i's initial experience at product j, (2) L_{ij}, which indicates the rate at which i learns how to produce product j, and (3) K_{ij} which indicates i's steady-state productivity rate at product j. We presume uncertainty in the parameters L_{ij}, which we model with a finite set of scenarios $\mathcal{S} = \{1, \ldots, S\}$.

Specifically, we associate to each scenario $s \in \mathcal{S}$ a probability $p(s)$, wherein $p(s) \geq 0 \ \forall s \in \mathcal{S}$ and $\sum_{s \in \mathcal{S}} p(s) = 1$. We then let L_{ij}^{s} denote the learning rate of individual i at product j in scenario $s \in \mathcal{S}$ and thus that individual's rate in a given period is calculated on a per-scenario basis as $r_{ij}^{\tau s}(\sum_{t=1}^{\tau} x_{ij}^{t}) = I_{ij} + K_{ij}[1 - e^{-\frac{\Sigma}{\cdot}}]$. Finally, we let the continuous variable o_{ij}^{ts} denote the amount the individual can produce of product j in period t in scenario $s \in \mathcal{S}$. Moreover, the inventory level of product $j \in \mathcal{J}$ in period $t \in \mathcal{T}$ in scenario $s \in \mathcal{S}$ is represented by the continuous variable b_j^{ts}. When demand for a product is greater than the amount in inventory, we let the continuous variable B_j^{ts} represent the number of items of product $j \in \mathcal{J}$, in period $t \in \mathcal{T}'$, in scenario $s \in \mathcal{S}$ that can not be provided. Figure 4.2 summarizes the sequence of decisions and events.

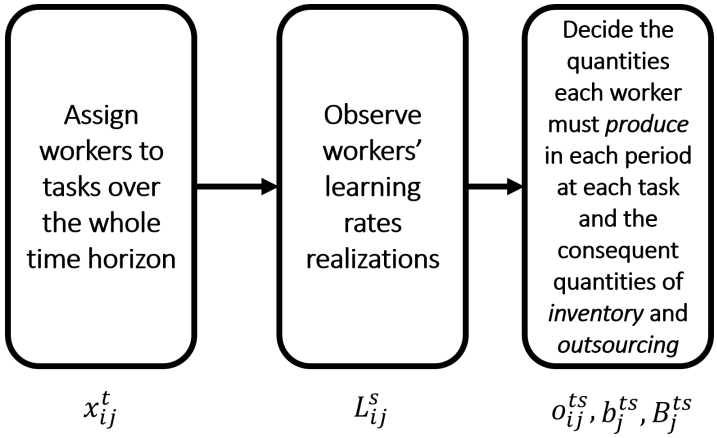


Figure 4.2: Illustration of the sequence of decisions and events

With these parameters and variables, we formulate the following mixed integer two-stage stochastic program with fixed recourse (see Birge and Louveaux [12] and Kall and Wallace [50]):

$$v_{SP} := \min \qquad \sum_{s \in \mathcal{S}} p(s) \{ \sum_{j \in \mathcal{J}} h_j \sum_{t \in \mathcal{T}} b_j^{ts} + \sum_{j \in \mathcal{J}} C_j \sum_{t \in \mathcal{T}} B_j^{ts} \} \tag{4.2}$$

subject to

$$\sum_{i\in\mathcal{I}} x_{ij}^{t} \leq 1 \qquad j \in \mathcal{J}, t \in \mathcal{T} \tag{4.3}$$

$$\sum_{j\in\mathcal{J}} x_{ij}^{t} \leq 1 \qquad i \in \mathcal{I}, t \in \mathcal{T} \tag{4.4}$$

$$b_{j}^{ts} = \sum_{i\in\mathcal{I}} o_{ij}^{ts} \qquad j \in \mathcal{J}, s \in \mathcal{S} \tag{4.5}$$

$$b_{j}^{ts} = \sum_{i\in\mathcal{I}} o_{ij}^{ts} + b_{j}^{(t-1)s} \qquad j \in \mathcal{J}, t \in \{\mathcal{T} \setminus \mathcal{T}' \setminus t_0\}, s \in \mathcal{S} \tag{4.6}$$

$$b_{j}^{ts} = \sum_{i\in\mathcal{I}} o_{ij}^{ts} + b_{j}^{(t-1)s} + B_{j}^{ts} - d_{j}^{ts} \qquad j \in \mathcal{J}, t \in \mathcal{T}', s \in \mathcal{S} \tag{4.7}$$

$$b_{j}^{(t-1)s} + B_{j}^{ts} \geq d_{j}^{t} \qquad j \in \mathcal{J}, t \in \mathcal{T}', s \in \mathcal{S} \tag{4.8}$$

$$o_{ij}^{ts} \leq r_{ij}^{ts} x_{ij}^{t} \qquad i \in \mathcal{I}, j \in \mathcal{J}, t \in \mathcal{T}, s \in \mathcal{S} \tag{4.9}$$

$$r_{ij}^{\tau s}(\sum_{t=1}^{\tau} x_{ij}^{t}) = I_{ij} + K_{ij}[1 - e^{-\frac{\Sigma_{-}}{-}}] \qquad i \in \mathcal{I}, j \in \mathcal{J}, t \in \mathcal{T}, s \in \mathcal{S} \tag{4.10}$$

$$x_{ij}^{t} \in \{0, 1\} \qquad i \in \mathcal{I}, j \in \mathcal{J}, t \in \mathcal{T} \tag{4.11}$$

$$o_{ij}^{ts} \in \Re_{+} \qquad i \in \mathcal{I}, j \in \mathcal{J}, t \in \mathcal{T}, s \in \mathcal{S} \tag{4.12}$$

$$r_{ij}^{ts} \in \Re_{+} \qquad i \in \mathcal{I}, j \in \mathcal{J}, t \in \mathcal{T}, s \in \mathcal{S} \tag{4.13}$$

$$b_{j}^{ts} \in \Re_{+} \qquad j \in \mathcal{J}, t \in \mathcal{T}, s \in \mathcal{S} \tag{4.14}$$

$$B_{j}^{ts} \in \Re_{+} \qquad j \in \mathcal{J}, t \in \mathcal{T}', s \in \mathcal{S}. \tag{4.15}$$

The objective function (4.2) calculates the expected cost associated with a set of production scheduling decisions. Constraints (4.3) and (4.4) enforce that each product in each period can be produced by at most one worker and that each worker in each period can produce at most one product. Constraints (4.5) fix the inventory level at the end of the first day of the month to be equal to the output produced in that day (since we assume that the initial inventory level is equal zero). Similarly, constraints (4.6) guarantee that, for periods different from the initial one and from the initial day of the week, the inventory level at the end of each day is equal to the sum of the total output of that period and the inventory level coming from the previous day. Relatedly, constraints (4.7) assure that, for the first day of each week, the inventory level is equal to what is produced in that period plus the quantity coming from the previous week (from production and outsourcing), after the demand satisfaction. Constraints (4.8) determine the amount of demand that is not met from inventory. Constraints (4.9) bound the amount an individual produces of a product in a period by their production rate, which is a function of their experience. We note that these constraints allow "practicing", meaning a worker can be assigned to a product without producing. Constraints (4.10) calculate each worker's productivity rate at each

product in each period as a function of their prior experience. Finally, constraints (4.11) to (4.15) define the variables and their domains.

This model, as presented, is non-linear (see constraints (4.9) and (4.10)). However, as the productivity rate function has a finite and discrete domain (see Hewitt, Chacosky, Grasman, and Thomas [44] (2015)), we can adapt their technique for representing such a function with binary variables and linear constraints, transforming model (4.2)-(4.15) into a mixed integer program. Details regarding the reformulation can be found in Appendix 3.a.

4.4.2 Production planning indicators

In this section, we discuss seven different indicators we calculate and analyze in order to determine how uncertainty in individual learning rates should impact production planning decisions. We run regression models to determine relationships between statistics calculated from instance parameter values and statistics calculated from optimal solutions of our linearized model (4.2)-(4.8), (3.a.1)-(3.a.6), (4.11)-(4.15).

Indicators based on instance parameter values We presume the organization has a belief regarding the distribution of individual i's learning rate, along with estimates of its mean, μ_i, and variance, σ_i^2. With these, we also calculate for each worker two indicators: the *Faster Than Average Probability* ($FTAP_i$) and the *Target Capacity Probability* (TCP_i).

In order to compute the $FTAP_i$, we first need to calculate the average estimated workforce learning rate, $\bar{\mu} = \sum_{i \in \mathcal{I}} \frac{\mu}{|\mathcal{I}|}$. Then, for each individual i, we set:

$$FTAP_i = p(L_i \leq \bar{\mu}) \qquad i \in \mathcal{I}, \tag{4.16}$$

i.e. the probability p that individual's learning rate L_i is no greater than the average $\overline{\mu}$ (and hence the individual learns at least as fast as the average worker).

For the TCP_i, we use a target capacity that is half of the individual's steady-state productivity rate, i.e. $K_i/2$. Then, for each worker i, we calculate the probability that their productivity is at least as great as $K_i/2$, when they produce a product each day of the first week. Specifically, given the distribution for i's learning rate, we set TCP_i equal to:

$$TCP_i = p(r_i^5(5) \geq K_i/2) \qquad i \in \mathcal{I}, \tag{4.17}$$

where 5 refers to the first five days of the week.

Indicators based on optimal solution values We presume that we have solved an instance of the stochastic programming model to (near-)optimality and have access to the resulting solution x_{ij}^{t*}. Having done so, we calculate two indicators.

The first indicator, na_i, measures the number of periods in which individual i is assigned to potentially produce a product and is given by:

$$na_i = \sum_{t \in \mathcal{T}} \sum_{j \in \mathcal{J}} x_{ij}^{t*} \qquad i \in \mathcal{I}. \tag{4.18}$$

The second indicator, np_i, is the number of different products, j, that individual i produces and it is given by:

$$np_i = \sum_{j \in \mathcal{J}: \exists t \in \mathcal{T}: x = 1} 1 \qquad i \in \mathcal{I}. \tag{4.19}$$

From the perspective of measuring cross-training, np_i, is incomplete, as it does not consider the number of times an individual produces items of a product. Thus, we also calculate the Coefficient-of-variation-type measure proposed by Valeva, Hewitt, Thomas, and Brown [88], which they refer to as the *Cross-training Index* (CTI). Specifically:

$$CTI_i = \frac{SD_i}{E_i}, \qquad i \in \mathcal{I}, \tag{4.20}$$

where SD_i is the standard deviation and E_i is the average of the number of periods worker i is assigned to a product. We note that larger values of CTI_i indicate greater specialization (an individual produces just a few products, but each many times), while smaller values indicate more cross-training (an individual produces many products, but each just a few times).

Similarly, we compute for each worker a practicing index, PI_i, which reflects the percentage of periods during which i is assigned to practice instead of production and it is calculated as:

$$PI_i = \frac{\sum_{j \in \mathcal{J}: \sum o = 0 \wedge \sum \sum o > 0} \sum_{t \in \mathcal{T}} x_{ij}^{t*}}{\sum_{j \in \mathcal{J}} \sum_{t \in \mathcal{T}} x_{ij}^{t*}} \qquad i \in \mathcal{I}. \tag{4.21}$$

Finally, as product sales prices (and hence the costs in our model) can vary, we also measure the average cost of the products produced by a worker. To do so, we calculate the average cost of the products produced by worker i by the following cost index:

$$CI_i = \frac{\sum_{j \in \mathcal{J}: \sum x > 0} (h_j + C_j)}{\sum_{j \in \mathcal{J}: \sum x > 0} x_{ij}^{t*}} \qquad i \in \mathcal{I}. \tag{4.22}$$

4.4.3 Linear regression models

We test the hypotheses presented in Section (4.3.2) with linear regression models (Stock and Watson [87] (2007)). For each instance, the set of Indipendent variables is composed of:

1. $\mu = [\mu_1, \ldots, \mu_I]$: vector of the means of the distribution of workers' learning rates,
2. $\sigma^2 = [\sigma_1^2, \ldots, \sigma_I^2]$: vector of the variances of the distribution ofs workers' learning rates,

3. $FTAP = [FTAP_1, \ldots, FTAP_I]$: vector of the probabilities that worker i learns faster than the average worker,

4. $TCP = [TCP_1, \ldots, TCP_I]$: vector of the probabilities worker i reaches half his/her steady state productivity when producing a product for a week,

5. $CI = [CI_1, \ldots, CI_I]$: vector of the average costs of items produced by worker i (this indicator is included only for testing hypotheses concerning the CTI_i and the PI_i).

We then seek to determine coefficients for the following linear models that will yield the best fit to the statistics derived from instance parameter values and optimal solutions to our stochastic program. We note that when fitting these equations, we only consider workers that were used at least once (i.e. $\sum_{j\in\mathcal{J}}\sum_{t\in\mathcal{T}} x_{ij}^{t*} \geq 1$). The first equation will help us test both Hypothesis 1 and 2:

$$\alpha_{\mu}^{na}\mu + \alpha_{\sigma}^{na}\sigma^2 + \alpha_{FTAP}^{na}FTAP + \alpha_{TCP}^{na}TCP + \beta^{na} = na. \tag{4.23}$$

The next equation will be used to test both Hypotheses 3 and 4:

$$\alpha_{\mu}^{np}\mu + \alpha_{\sigma}^{np}\sigma^2 + \alpha_{FTAP}^{np}FTAP + \alpha_{TCP}^{np}TCP + \beta^{np} = np. \tag{4.24}$$

The next equation will be used to test Hypothesis 5:

$$\alpha_{\mu}^{CTI}\mu + \alpha_{\sigma}^{CTI}\sigma^2 + \alpha_{FTAP}^{CTI}FTAP + \alpha_{TCP}^{CTI}TCP + \alpha_{CI}^{CTI}CI + \beta^{CTI} = CTI. \tag{4.25}$$

Finally, the next equation will be used to test Hypothesis 6:

$$\alpha_{\mu}^{PI}\mu + \alpha_{\sigma}^{PI}\sigma^2 + \alpha_{FTAP}^{PI}FTAP + \alpha_{TCP}^{PI}TCP + \alpha_{CI}^{PI}CI + \beta^{PI} = PI. \tag{4.26}$$

Before fitting these equations to data and analyzing the statistical significance of the coefficients to test our hypotheses, we normalize the Indipendent variables values as suggested in Gelman [36] (2008). Specifically, each Indipendent variable value has been mean centered and divided by two standard deviation. As CTI and PI can be computed only for assigned workers, for the last two regressions, the normalization was done based only on the values for assigned workers.

4.5 Experimental setting

In this section, we describe how the instances for our problem are generated. Subsection 4.5.1 describes the different parameter values related to the production setting, while Subsection 4.5.2 presents how uncertainty in worker learning rates is represented. Finally, Subsection 4.5.3 explains how to determine the number of instances to consider.

We note that all computational experiments were run on a computer with 8 GB of RAM and a 2.70 GHz CPU. All software was implemented in Python 3.6.1 (Python Software Foundation [74]) with Gurobi 7.5.1 (Gurobi [38]) to its default optimality tolerance. Regression equations were fitted with the Python package statsmodels Statsmodels-developers [86].

4.5.1 Instance parameter values

Since we want to draw insights on how workers' characteristics influence the assignment decisions, we consider instances where there are more workers than products and more periods than workers ($| \mathcal{J} | < | \mathcal{I} | < | \mathcal{T} |$). Specifically, in our experiments we consider a working month ($| \mathcal{T} |$=20 days), six workers ($| \mathcal{I} | = 6$) and three products ($| \mathcal{J} | = 3$). We have $\gamma = 5$ (days), and $\mathcal{T}' = \{5, 10, 15, 20\}$.

We assume that each product j has a different *cost-of-goods-sold (*$COGS_j$*)*: $COGS_1 = 100$, $COGS_2 = 300$, $COGS_3 = 500$. These values of $COGS$ are used to compute inventory and unmet demand costs. Specifically, annual inventory costs are assumed to be equal to the 24% of COGS (see Mazzola, Neebe, and Rump [63] (1998)). As we assume a year consisting of 240 working days and we are interested in obtaining daily inventory costs, we apply the formula $\frac{0.24COGS}{240}$, thus obtaining $h_1 = 0.1, h_2 = 0.3, h_3 = 0.5$. Regarding the cost of unmet demand, we consider two different values for each product, one that is 110% of COGS and one that is 125% of COGS. Thus, we first consider outsoucing costs $C_1 = 110$, $C_2 = 330$, $C_3 = 550$ and, then, $C_1 = 125$, $C_2 = 375$, $C_3 = 625$.

Regarding product demands d_j^t, as we presume COGS vary by product, to limit the number of changing parameter values in our instances, we presume that each product has the same demand, d, in each week. Formally, we presume that $d_j^t = d_j^t = d, \forall j \in \mathcal{J}, \forall t, t' \in \mathcal{T}$ and $d_j^t = d_j^t = d, \forall j, j' \in \mathcal{J}, \forall t \in \mathcal{T}$. We randomly draw two demand values ($d1$ and $d2$) from the discrete interval [1,500], yielding two groups of instances, wherein instances in different group have different demands, but instances in the same group do not. However, instances in the same group do differ in the parameters of individual worker learning curves.

Regarding the parameters values of individual worker learning curves, as we focus on when a company is making new products, we set each worker's initial experience on all products to zero (i.e. $I_{ij} = 0, i \in \mathcal{I}, j \in \mathcal{J}$). We also presume that each worker has the same steady-state productivity rate for all products. In addition, to focus on the rate at which individuals learn, we set the steady-state rate for all individuals to the same value. Specifically, we set $K_{ij} = 4d, \forall i \in \mathcal{I}, j \in \mathcal{J}$. Regarding individual learning rates, we presume that all worker learning rates follow a truncated normal distribution, but that these distributions differ in their parameter values. As a baseline, we consider the distribution parameter values given in Table 4.1 for each worker.

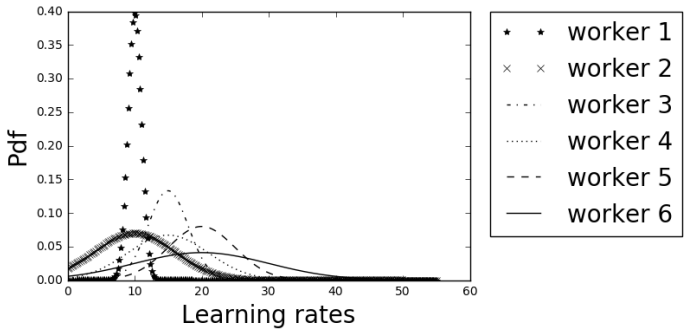


Figure 4.3: Learning rate distributions

Worker	μ_i	σ_i
1	10	1
2	10	6
3	15	3
4	15	6
5	20	5
6	20	10

Table 4.1: Parameter values

Furthermore, we assume that the learning rates across workers are not correlated and, consequently, they are indipendently drawn.

Then, to assess the impact of variance in learning rates, we consider three different standard deviations levels for each worker, σ_i, $1.25\sigma_i$, and $1.5\sigma_i, \forall i \in \mathcal{I}$, where σ_i refers to worker i standard deviation in Table 4.1. We summarize the different instance factors and their levels in Table 4.2. With these three factors, and their associated levels, we have twelve different combinations of factor levels (see figure 4.4).

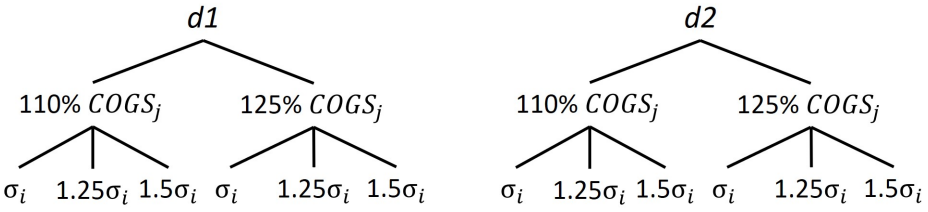


Figure 4.4: Different factor levels combinations

Factor	Levels
Outsourcing cost	110%, 125% of COGS
Product demand	Two values drawn from $[100, 500]$
Learning rate standard deviation	$1\sigma, 1.25\sigma, 1.5\sigma$

Table 4.2: Instance factors

4.5.2 Size of the scenario tree

Our analysis in the next section is based on solving instances of a stochastic program wherein statistical distributions are represented with scenarios. As such, it is important that we construct a set of instances so that we can be confident, statistically-speaking, that our findings are generally true, and not an artifact of the instances used. Thus, we undertook the following formal procedures to determine both how many scenarios to include in an instance and how many instances to generate.

To ensure that we have the appropriate number of scenarios in an instance, we performed an in-sample and out-of-sample stability analysis using the procedure presented in Kaut and Wallace [51] (2007). Both types of analysis seek to determine the number of scenarios that are needed to obtain a stable optimal objective function value. For the in-sample analysis, we solved the stochastic program for each of the twelve combinations of factor levels and six scenario trees of increasing size. Thus, this analysis is based on solving 72 instances of the stochastic program, and we report the average of the twelve objective function values for a given number of scenarios in Table 4.3. Even if the objective function value looks very stable, we consider the scenario tree composed of 500 scenarios the benchmark for our experiments. Furthermore, we conclude from these results that 100 scenarios may be sufficient for the stochastic program to yield an objective function value that is close to the value one could achieve if the stochastic program considered the full distribution, and not a set of scenarios.

	$\mid \mathcal{S} \mid$=10	$\mid \mathcal{S} \mid$=50	$\mid \mathcal{S} \mid$=100	$\mid \mathcal{S} \mid$=200	$\mid \mathcal{S} \mid$=300	$\mid \mathcal{S} \mid$=500
Average obj. func. value (over 12 instances)	373.02	373.72	373.61	373.67	373.59	373.68

Table 4.3: In-sample analysis results

Out-of-sample stability analysis gives another perspective on the question of how many scenarios to generate, as it attempts to measure how a solution produced when representing a distribution by a scenario tree of a smaller size will perform in the "real world", and the full distribution is experienced.

To perform this analysis, we generated a benchmark set of 500 scenarios to serve as a proxy for the full distribution. We then took the solutions produced by solving the previously discussed 72 instances of the stochastic program and evaluated them with respect to these 500 scenarios. We report the average increase in objective function value, by number of scenarios, in Table 4.4. We again see that scenario trees of size 100 yields a fairly stable objective function value. Finally, considering that solving our stochastic program with 100 scenarios required 77.60 CPU seconds, on average, while solving the stochastic program with 500 scenarios required 1232.32 CPU seconds, on average, the regression analysis in section 4.6.2 is based on instances with 100 scenarios.

	$\mid \mathcal{S} \mid$=10	$\mid \mathcal{S} \mid$=50	$\mid \mathcal{S} \mid$=100	$\mid \mathcal{S} \mid$=200	$\mid \mathcal{S} \mid$=300	$\mid \mathcal{S} \mid$=500
Average obj. func. gap (over 12 instances)	0.089 %	0.019%	0.007%	0.015%	0.013%	0.00%

Table 4.4: Out-of-sample analysis results

4.5.3 Number of instances

In this subsection, we present the method used to determine the number of instances to be considered for each combination of factor levels. As we intend to analyze coefficients from a linear regression model to derive managerial insights (see subsection 4.4.3), we determine the sample size required to be statistically confident that a correlation coefficient is different from zero. Specifically, we compute the sample size N through the formula (Hulley, Cummings, Browner, Grady, and Newman [46] (2013)):

$$N = [\frac{Z_\alpha + Z_\beta}{C}]^2 + 3, \tag{4.27}$$

where N is the total sample size (i.e. the number of instances which have to be generated for each combination), Z_α is the standard normal deviate for type I error rate, Z_β is the standard normal deviate for type II error rate and $C = 0.5 \ln[\frac{(1+\rho)}{(1-\rho)}]$, where ρ is the expected correlation coefficient. We substitute the parameter values $\alpha = .05, \beta = .05, \rho = .7$ in the formula and we obtain that we should create 20 instances for each factor level combination (see figure 4.4). In summary, for each of the twelve combinations of factor levels we generate 20 instances by sampling different scenario trees of the same size ($\mid \mathcal{S} \mid = 100$) from the distributions (see figure 4.3 and table 4.1) discussed above. Thus, much of our analysis is based on a set of 240 instances.

4.6 Results and analysis

In this section, we first assess the value in recognizing uncertainty in worker learning rates when developing production plans. We then report results related to fitting the regression equations presented in Section 4.4.3. We finally discuss how those results relate to the hypotheses presented in Section 4.3.2 regarding how uncertainty in learning should be recognized.

4.6.1 Value in modeling uncertain learning rates

We next assess the value in explicitly modeling uncertainty in worker learning rates. To do so, we consider instances for each of the 12 combinations of factor levels and based on the benchmark scenario tree with 500 scenarios discussed above. We then calculate averages of the following metrics:

1. *Relative Expected Value of Perfect Information (%EVPI)* (Birge and Louveaux [12] (2011)),
2. *Relative Value of the Stochastic Solution (%VSS)* (Birge and Louveaux [12] (2011)),
3. *Relative Loss of Using the Skeleton Solution (%LUSS)* (Maggioni and Wallace [56] (2012)),
4. *Relative Loss of Upgrading the Deterministic Solution (%LUDS)* (Maggioni and Wallace [56] (2012)).

To compute the *%EVPI*, we compare the objective function value of the solution to the stochastic program, v_{SP}, with the expected objective function value, v_{WS}, of the problem where we assume we have a full knowledge of workers' learning rates already in the first stage (*Wait and See (WS)* problem). The *%EVPI* is then defined as the relative gap between these objective function values, that is:

$$\%EVPI = \frac{v_{WS} - v_{SP}}{v_{SP}}. \tag{4.28}$$

The average *%EVPI* over the 12 combinations of factor levels is -0.28%, suggesting that the potential savings associated with postponing workers assignment decisions until their production rates are revealed are limited.

To compute the *%VSS*, we solve the *Expected Value (EV)* problem, where workers' learning rates correspond to their mean values. Then, the stochastic program is solved with first-stage decision variables fixed to their values obtained in the solution of the *EV* problem. The expectation of the objective function value v_{EEV}, of this problem is referred to as the *Expected Result of using the EV solution (EEV)*. The *%VSS* is then the relative gap, computed as follows:

$$\%VSS = \frac{v_{EEV} - v_{SP}}{v_{SP}}. \tag{4.29}$$

The average *%VSS* over the 12 combinations of factor levels is 2.78%, suggesting that the savings associated with solving the stochastic program instead of the deterministic one are valuable.

We next analyze to what extent we can derive information from the *EV* solution to obtain a high quality solution of the *SP*. The *%LUSS* metric estimates the degree by which a solution to the *EV* problem prescribes a subset of assignment choices that correspond to those in the solution to the stochastic program. To calculate this metric, we solve the *Expected Skeleton Solution Value (ESSV)* problem, which is the stochastic program wherein a worker cannot work on a product in a period if the solution to the *EV* problem does not allow him/her to do so. Denoting the objective function value of this stochastic program as v_{ESSV}, we calculate the *%LUSS* as:

$$\%LUSS = \frac{v_{ESSV} - v_{SP}}{v_{SP}}. \tag{4.30}$$

The average *%LUSS* over the 12 combinations of factor levels is 2.70%, suggesting that the main reason of a bad deterministic solution is due to the wrong schedule of workers' assignments.

Relatedly, the *%LUDS* metric estimates the degree by which a solution to the *EV* problem is *upgradeable* in a stochastic environment. To calculate this metric, we solve the *Expected Input Value (EIV)* problem, which is the stochastic program wherein each worker must be assigned to a task in at least as many periods as prescribed by the *EV* solution. Denoting the objective function value of this stochastic program as v_{EIV}, we calculate the *%LUDS* as:

$$\%LUDS = \frac{v_{EIV} - v_{SP}}{v_{SP}}. \tag{4.31}$$

The average *LUDS* over the 12 combinations of factor levels is 1.60%, suggesting that the deterministic solution can be *upgraded* only for a subset of assignments decisions.

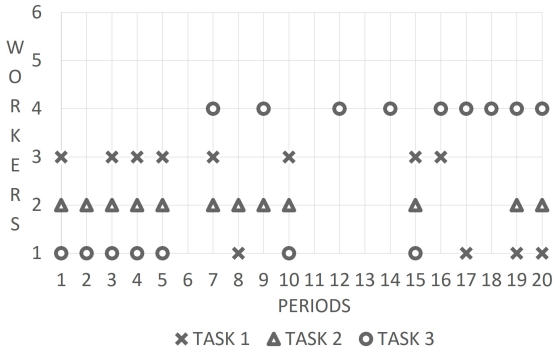
Fundamentally, the first stage of the stochastic program determines which products and which period a worker should work on. Thus, we continue our analysis in order to determine whether the solution to the *EV* assigns workers to the wrong products, to the wrong periods, or both. Regarding products, we solve a variant of the stochastic program wherein a worker is assigned to the same products he/she is assigned to in the solution to the *EV*, leaving the stochastic program to determine when a worker performs those products. We calculate a relative gap in objective function values like those above and see a degradation of only 0.13%. Conversely, we also solved a variant of the stochastic program wherein a worker is constrained to only work in the same periods he/she is assigned to in the solution to the *EV*, leaving the stochastic program to determine which products a worker performs in each of those periods. Here, the relative gap in objective function values was 1.00%. We conclude from these relative gaps that the solution to the *EV* problem often assigns workers to the right products, but not necessarily in the right periods.

Figure 4.5 displays the assignment schedules for the previous experiments for one instance. Comparing figures 4.5a and 4.5b, we notice that the two assignments schedules are completely different and that in the deterministic solution one additional worker is required. Furthermore, looking at figures 4.5b and 4.5c, the plans are identical, meaning that the stochastic solution starting from the deterministic one does not change any assignment variables. Moreover, referring to figures 4.5a and 4.5d, we notice that the deterministic solution is not upgreadable, as there are multiple assignments in the deterministic solution which are null in the *SP* one and viceversa. The comparison between figures 4.5b and 4.5d also shows that, in the *EIV* solution, worker 2 spends more periods on task 3. We also notice that in figure 4.5e, only periods to which a worker is assigned to change, not the tasks, while in figure 4.5f, if a worker is not assigned in a period in the deterministic solution, it is neither assigned to that period in the *ESSV* with fixed periods solution. Finally, it is interesting to observe that by fixing the tasks and/or the periods suggested by the deterministic solution, the assignment plan admits a limited level of cross-training. As a matter of fact, in both plans, only worker 1 cross-trains, which is a very similar situation to the one suggested by the *SP* solution.

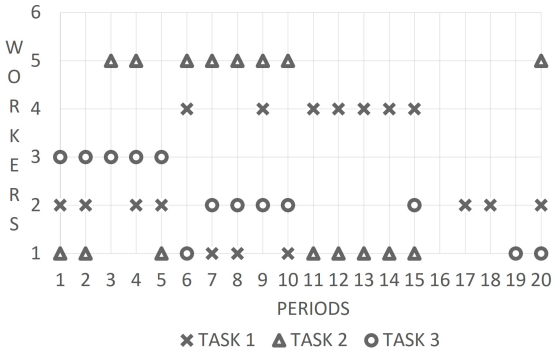
Finally, we turn our attention to how solutions to the stochastic and deterministic problems differ with respect to the timings of scheduling and production decisions. Specifically, in Figure 4.6a, we illustrate for solutions to the stochastic and deterministic problems, averages of the first day within each week during which some worker is scheduled to potentially produce. We see that, on average, the stochastic program schedules potential production later in the week than the deterministic model. In addition, the first period is later in each succeeding week. Recalling that some of these scheduled production assignments may in fact be to practice, we hypothesize that by considering the full distributions of learning rates (through scenarios), the stochastic program has a better estimate of the impact of practice than the deterministic model and thus schedules less. In Figure 4.6b, we report similar averages, only of the first period when production begins. We recall that production decisions are made in the second stage and, thus, they depend on scenarios. Thus, we use the solution to the *EEV* problem, wherein production decisions are made for each scenario based upon the scheduling decisions from the deterministic problem. We see that production occurs earlier given the schedule prescribed by the deterministic model, which in turn leads to higher inventory costs.

4.6.2 Regression results for hypotheses testing

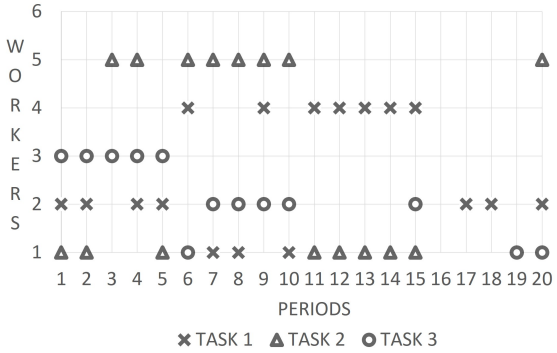
We turn our attention to characterizing how uncertainty in learning should impact decisions related to worker assignment, cross-training, and practicing. In order to test the hypotheses described in subsection (4.3.2), we first report in Table 4.5 the results of fitting the regression equations presented in Section 4.4.3 to indicators (see subsection 4.4.2) derived from the 240 instances of the stochastic



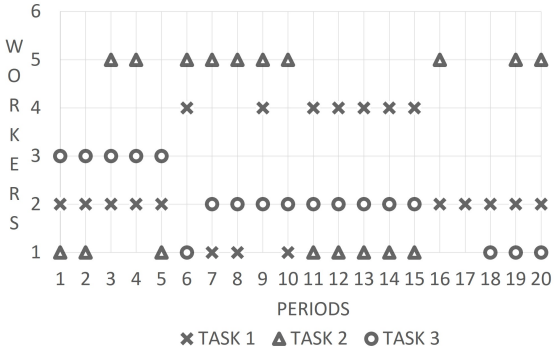
(a) RP solution



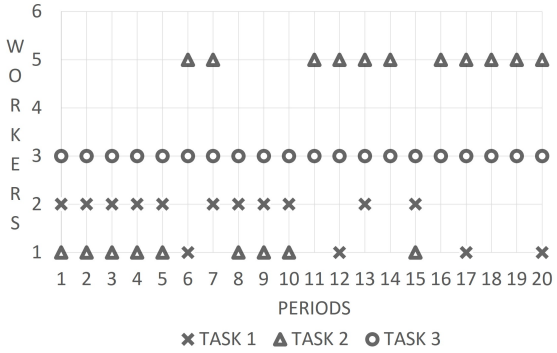
(b) EEV solution



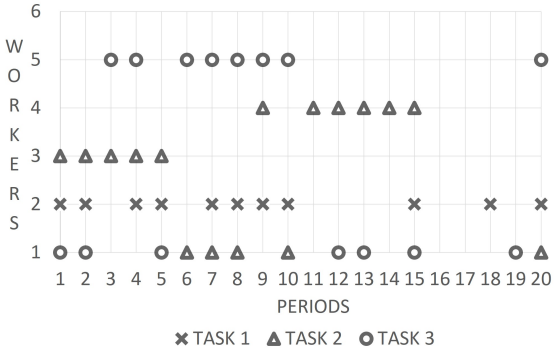
(c) ESSV solution



(d) EIV solution

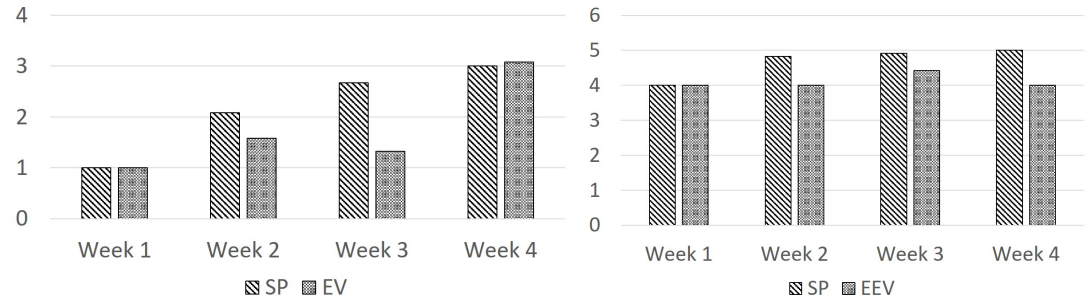


(e) ESSV fixed task solution



(f) ESSV fixed period solution

Figure 4.5: Stochastic solution analysis for an instance



(a) Average first scheduled period **(b) Average first production period**

Figure 4.6: Comparing scheduling and production decisions

program with 100 scenarios, as well as near-optimal solutions to those stochastic programs. Detailed results can be found in Appendix 3.b.

Dep.		β		μ		σ		$FTAP$		TCP		CI	
var.	r	α	p-value	α	p-value	α	p-value	α	p-value	α	p-value	α	p-value
na (eq.(4.23))	0.627	6.1597	0.000	-6.0808	0.000	-0.9677	0.000	-12.9385	0.000	14.3039	0.000	N/A	N/A
np (eq.(4.24))	0.579	0.6931	0.000	-0.1340	0.025	-0.3255	0.000	-0.5788	0.000	1.2190	0.000	N/A	N/A
CTI (eq.(4.25))	0.278	1.5345	0.000	0.0616	0.212	0.1653	0.000	0.0202	0.695	0.0799	0.311	-0.0013	0.000
PI (eq.(4.26))	0.282	0.4912	0.000	-0.0266	0.274	0.0303	0.043	-0.0012	0.964	-0.1431	0.000	0.0008	0.000

Table 4.5: Summary regression results

From table 4.5, we note that the first two regression equations (see equations (4.23) and (4.24)) have reasonably high r^2 values, suggesting the linear models fit well to the data. In addition, all the p-values for the coefficients in these fitted regression models are less than 5%, suggesting we can derive insights from those coefficients that we can be confident in, statistically-speaking. While the r^2 values indicate that the second two equations (see equations (4.25) and (4.26)) do not fit the data as well, they are high enough that the coefficients can still be analyzed to yield insights.

Recall that we normalized the values of the statistics in the data sets to which these equations were fitted so that we may directly compare these coefficients (see Gelman [36] (2008)). We first note that for the first two regression equations, the coefficients associated with statistics derived from the distributions of worker learning rates ($FTAP, TCP$) are larger in absolute value than those associated with the parameter values of those distributions. We view this as validation that there is more predictive power in statistics that represent the entire distribution than the values of individual parameters. We next examine these coefficients in the context of the previously presented hypotheses to derive managerial insights.

4.6.3 Which types of workers should produce more often?

Hypotheses 1 and 2 (see subsection (4.3.2)) relate to identifying which types of workers should produce in more periods. The first hypothesizes that workers who are more likely to reach a target capacity (i.e. have a higher TCP) should work more periods while the second hypothesizes that workers that are less likely to be faster than average learners (i.e. have a lower $FTAP$) should work more periods. We assess the validity of these hypotheses by examining the first regression equation, wherein the dependent variable is na_i. We illustrate in Figure 4.7 the coefficients associated with all four independent variables ($\mu, \sigma^2, FTAP, TCP$), as they all have a p-value that is less than 5%, and thus we can be statistically confident that all four have predictive power regarding which types of workers should work more periods.

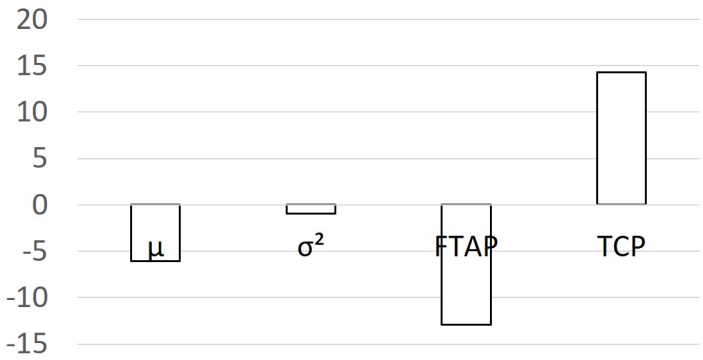


Figure 4.7: Coefficients for na regression.

Recalling that the independent variable values were normalized to be on the same scale, we see that the $FTAP$ and TCP are the leading indicators regarding which workers should work the most periods. The coefficient, α^{na}_{TCP} is largest in absolute value, and positive, suggesting that workers that are more likely to reach their target capacity should work the most, verifying our hpothesis 1. The next largest coefficient in absolute value is α^{na}_{FTAP}. However, this coefficient is negative, suggesting that workers with lower $FTAP$, meaning they are less likely to learn faster than the average member of the workforce, should work more periods, verifying our hypothesis 2. Put another way, this suggests that those who are likely to be slower than average learners should work more. While the other two independent variables (μ, σ^2) also have p-values less than 5%, their coefficients are much smaller (in absolute value), and thus we view the $FTAP$ and TCP as the leading indicators regarding which workers should work the most periods.

4.6.4 Which types of workers should produce more products?

The next two hypotheses relate not to which types of workers should work on more products. Hypothesis 3 posits that workers for whom the organization is more confident in their estimate of their learning rate (i.e. σ is smaller) should work on more products. Hypothesis 4 also relates the TCP to the number of products a worker should make. We assess the validity of these hypotheses by examining regression equation (4.24), wherein the dependent variable is np. We illustrate in Figure 4.8 the coefficients associated with all four independent variables, as they all have a p-value that is less than 5%. We again see that the coefficient associated with TCP, α^{np}_{TCP}, is largest in absolute value, and

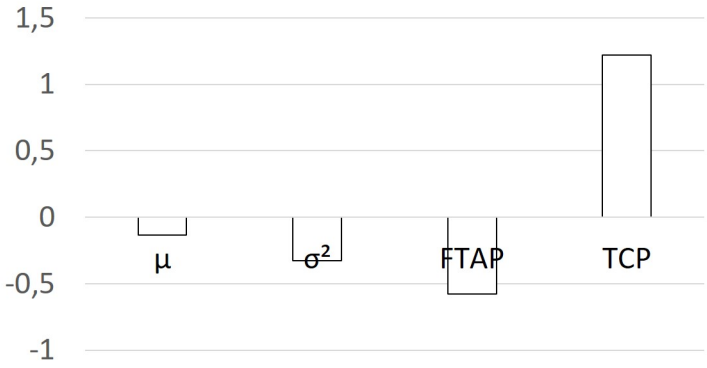


Figure 4.8: Coefficients for np regression.

positive, suggesting that workers that are more likely to reach their target capacity should work on more products. This verifies our hypothesis 4. Next in absolute value is the coefficient associated with $FTAP$, α^{np}_{FTAP}, which is negative, suggesting that workers who are likely to be slower than average learners should work on more products. Finally, the coefficient associated with σ^2, (α^{np}_{σ}) is third-largest in absolute value. It is also negative, suggesting that the smaller the variance in the distribution of a worker's learning rate, the more products he/she should be assigned to produce. While we view this as verifying our hypothesis 3, comparing the coefficients associated with TCP and σ^2 in absolute value, we see the TCP indicator is the leading one of which workers should work on more products.

4.6.5 Which types of workers should cross-train and which should practice?

Hypothesis 5 relates to identifying which types of workers should cross-train more, whereas hypothesis 6 posits which types of workers should practice more. We assess hypothesis 5 with regression equation (4.25), which has CTI as the dependent variable and hypothesis 6 with equation (4.26), which has PI_i as the dependent variable. We illustrate the coefficients with p-value that is less than 5% for each equation in Figure 4.9. Recall that the higher the value of CTI, the less a worker cross-trains.

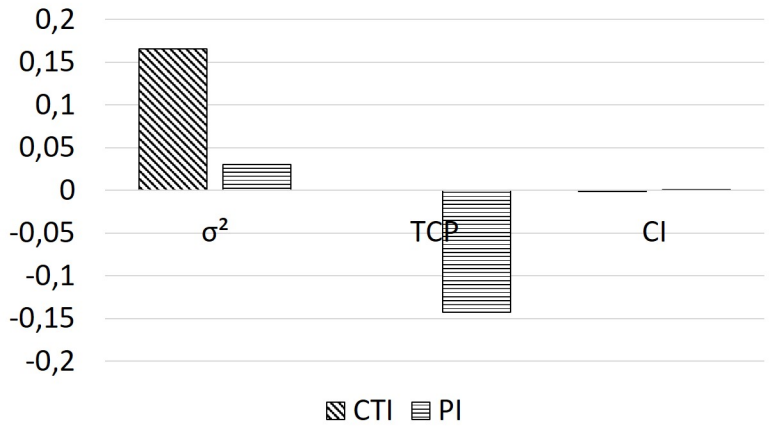


Figure 4.9: Coefficients for CTI, PI regressions with p-value $< .05$

For both dependent variables, we see that while we can have statistical confidence that the cost index coefficients, $\alpha_{CI}^{CTI}, \alpha_{CI}^{PI}$ are not zero, they are very small in magnitude. Hypothesis 5 posits that workers for whom the organization has less confidence regarding their learning rate (i.e. the variance of their distribution is larger) should cross-train less. Observing that α_{σ}^{CTI} is positive and has a p-value of 0.000 in the regression equation with CTI as the dependent variable, we conclude that this hypothesis is verified. This also corroborates our validation of hypothesis 4, where we saw that workers with smaller variances should work on more products. Finally, hypothesis 6 theorizes that workers with lower TCP should practice more. Observing that the α_{TCP}^{PI} is negative and has a p-value of 0.000 in the regression equation with PI as the dependent variable, we conclude that this hypothesis is verified as well, which is not surprising given that we have already observed that those with high TCP should produce more often.

4.6.6 Tactics for accommodating uncertainty in worker learning rates

In this section, we summarize the previous results and analysis in order to present tactics that an organization can use to accommodate uncertainty in worker learning rates when deriving production plans. We first conclude that an organization should recognize that there can be uncertainty in worker learning rates, as doing so can reduce costs. Regarding how to recognize uncertainty when assigning work, an organization should prioritize workers based on a measure that captures their estimations of the distributions of worker learning rates, such as the TCP, instead of individual distribution parameter values. Relatedly, an organization should prioritize the hiring of workers that have a high probability of quickly reaching a target capacity. Finally, confidence regarding the estimation of worker learning rates should impact cross-training decisions. Specifically, the organization should prioritize

cross-training workers for whom they have a strong belief that the estimated learning rate is the actual.

4.7 Conclusions and future works

In this paper, we considered a product manufacturer that makes multiple products and has complete information regarding customer demand timings and quantities for those products. In addition to recognizing that human learning occurs, and thus worker productivity as a function of experience can be predicted with a learning curve, the organization also recognizes that they do not have complete information regarding the parameters of that curve. Specifically, they do not know with certainty the rates at which workers learn, and hence their productivity. Thus, while demands are known with certainty, supply is not, and this manufacturer wants to derive a production plan that will minimize expected cost while meeting those demands. While the learning curve we consider in this paper is non-linear, we adapted a reformulation technique from the literature to formulate a stochastic programming model of this production planning problem as a mixed integer linear program, and we show that it can be solved quickly with today's solvers.

We used this stochastic programming model as the basis of an extensive computational study into whether and how uncertainty in worker learning rates should be recognized in production planning. To ensure confidence, statistically-speaking, in the insighs derived from this computational study, we based it on a set of instances generated with established techniques from the literature. We first observed that a manufacturer can reduce their costs by explicitly recognizing uncertainty in worker learning rates. To capture that uncertainty with a single measure by which workers can be ranked, we presented statistics based on the distributions of worker learning rates. We saw in our computational study that these statistics have strong predictive power regarding which types of workers should work more and on more products. We also derived insights from this computational study into the role that uncertainty in learning should play in cross-training decisions.

Regarding future works, the clear next extension is to model uncertainty in demands, particularly quantities. Doing so would enable study of the interplay between uncertainty in human learning (supply) and demand. In this case, an additional effort consisting in analyzing the reduced costs of the out-of-basis variables in the deterministic solution (see Crainic, Maggioni, Perboli, and Rei [25] (2017) for details) would be valuable in order to decrease the computational complexity. Another possible extension is to consider that an individual's learning rate can be more precisely estimated through testing, instead of considering means and variances of learning rate distributions. Consequently, we could include the choice of whether and which workers should be tested reducing the variance of the learning rate distribution. Finally, future works could include a multistage stochastic program

formulation for a sequential workforce assignment decision problem, wherein productivities (and hence learning rates) are observed in time, and schedules consequently updated.

Chapter 5

Optimization driven monotonic bounds for two-stage stochastic integer programs

Authors: Francesca Maggioni[1], **Rossana Cavagnini**[2]

Keywords: Two-stage stochastic integer programming, Bounds, Refinement chain

[1]University of Bergamo, Via dei Caniana, 2, Bergamo, Italy, e-mail: francesca.maggioni@unibg.it
[2]University of Bergamo, Via dei Caniana, 2, Bergamo, Italy, e-mail: r.cavagnini@studenti.unibg.it

5.1 Introduction

The aim of this chapter is to propose an approach which allows to obtain an optimized chain of lower bounds for a two-stage stochastic program by solving subproblems based on scenario grouping. In Ryan, Ahmed, Dey, and Rajan [82] (2016), the authors propose a model which allows to build the optimal scenario grouping given a maximum group cardinality. In Maggioni and Pflug [55] (2016), the authors propose monotonic chains of lower bounds to the optimal stochastic program based on the concavity of the probability mapping typical of stochastic programs, including those of the expectation type. However, the refinement chains of lower bounds are obtained choosing groups of scenarios of a given cardinality randomly, not allowing to guarantee that the chosen grouping provides the best lower bound.

In this chapter, we provide an optimized monotonic chain of lower bounds, by combining the properties of the probability mapping of stochastic programs presented in Maggioni and Pflug [55] (2016) with the methodology presented in Ryan, Ahmed, Dey, and Rajan [82] (2016). Consequently, we are able to construct a sequence of scenario groupings such that the estimated improvement with respect to the solution of single scenario problems is maximized. For this reason, we propose a mixed integer linear program to build monotonic chains of lower bounds, which allows to group scenarios disjointly. Through our computational results, we compare the lower bounds obtained through our approach with the ones obtained through the methodologies proposed by Maggioni and Pflug [55] (2016) and Ryan, Ahmed, Dey, and Rajan [82] (2016).

The chapter is organized as follows. Section (5.2) includes the preliminaries to our work and the mixed integer program for scenario grouping. Section (5.3) presents the computational results. Finally, in Section (5.4) some conclusions are reported.

5.2 Scenario grouping in refinement levels

In this section, we first present the notation and the preliminaries to our work and then, we formulate a mixed integer program to build scenarios groupings in order to obtain the optimized monotonic chain of lower bounds.

5.2.1 Notation and Preliminaries

We consider the following two-stage stochastic programming problem:

$$\min_x \{\mathbf{E}_\Pi[f(x,\xi)] : x \in \mathcal{X}\}, \tag{5.1}$$

where x represents the solution vector in the feasible region $\mathcal{X}$. The random vector ξ is defined on a probability space with support Ξ and the expectation $\mathbf{E}_\Pi$ is taken with respect to the probability Π, while f represents the objective function. Moreover, we suppose that Ξ is finite and it is represented by a set of $\mathcal{K} = \{1, \ldots, K\}$ scenarios. The problem can be then re-stated as:

$$z^* = \min_x \{\sum_{k \in \mathcal{K}} \pi_k f_k(x) : x \in \mathcal{X}\}, \tag{5.2}$$

where π_k and f_k are the probability and the objective function, respectively, associated with scenario k.

Following the approach presented in Ryan, Ahmed, Dey, and Rajan [82] (2016), we re-write (5.2) making the nonanticipativity constraints $\sum_{k \in \mathcal{K}} A_k x_k$ explicit:

$$z^* = \min_{x,\ldots,x} \{\sum_{k \in \mathcal{K}} \pi_k f_k(x_k) : x_k \in \mathcal{X}\ \forall k \in \mathcal{K}, \sum_{k \in \mathcal{K}} A_k x_k = 0\}, \tag{5.3}$$

where the non-anticipativity constraints force the variables x_k to be the same across scenarios. By dualizing the nonanticipativity constraints with multipliers λ, we write the following nonanticipative relaxation:

$$z^*(\lambda) = \sum_{k \in \mathcal{K}} \pi_k z_k^*(\lambda) \qquad \text{where} \qquad z_k^*(\lambda) = \min_x \{f_k(x) + \lambda^T A_k x : x \in \mathcal{X}\}\ \forall k \in \mathcal{K}. \tag{5.4}$$

By doing so, we obtain $z^*(\lambda)$ which is a lower bound on z^*, and the dual problem of maximizing $z^*(\lambda)$ with respect to λ returns the greatest lower bound.

Maggioni and Pflug [55] (2016) present two methods for building refinement chains. The first one consists in building disjoint scenario groups, while the second one consists in building scenario groups by keeping one or several scenarios fixed across all groups. For both cases, they show that by using the concavity of the probability mapping typical of the stochastic programs with expectation, a chain of lower bounds can be obtained.

Since we are interested in building a chain of lower bounds, we need to define the set of chain levels by $\mathcal{P} = \{1, \ldots, P\}$. Furthermore, for each chain level $p \in \mathcal{P}$, we can build lower bounds, by considering a set of groups $\mathcal{M}^p = \{m_1^p, \ldots, m_M^p\}$.

The refinement chain can be represented as:

$$\begin{aligned} \mathcal{M}^P: &\qquad \{m_1^P\} = \mathcal{K} \\ &\qquad \vdots \\ \mathcal{M}^p: &\qquad \{m_1^p, m_2^p, \ldots, m_M^p\} \\ &\qquad \vdots \\ \mathcal{M}^2: &\qquad \{m_1^2, m_2^2, \ldots, m_M^2\} \\ \mathcal{M}^1: &\qquad \{m_1^1, m_2^1, \ldots, m_M^1\} = \{\{k_1\}, \{k_2\}, \ldots, \{k_K\}\} \end{aligned} \tag{5.5}$$

where each row is a collection of groups of the set of scenarios $\mathcal{K}$ with the property that their union covers the whole space $\mathcal{K} = \bigcup_{l=1}^{M} m_l^p$ for all p and that each group m_l^p is the union of sets from the next more refined collection,

$$m_l^p = \bigcup_{m \subseteq m} m_s^{(p-1)}.$$

Figure 5.1 illustrates one of the possible refinement chains in the case of disjoint groups.

For details on the probability definition of each group we refer to Maggioni and Pflug [55] (2016).

Without loss of generality, from this point, we will denote a group in a refinement level simply by $m \in \mathcal{M}^p$. Furthermore, we represent the set of scenarios in each group of each refinement level by $\mathcal{K}^{p,m}$. For the disjoint case, at each refinement level p, the set of scenarios $\mathcal{K}$ is partitioned into M^p groups $(\mathcal{K}^{p,1}, \ldots, \mathcal{K}^{p,M})$ such that:

$$\bigcup_{m \in M} \mathcal{K}^{p,m} = \mathcal{K},\ \forall p \in \mathcal{P} \text{ and } \mathcal{K}^{p,m} \cap \mathcal{K}^{p,n} = \emptyset, \forall m, n \in \mathcal{M}^p, \forall p \in \mathcal{P}.$$

Thanks to this construction (see Maggioni and Pflug [55] (2016)), it is possible to show that the following chain of lower bounds to the optimal solution is:

$$z^{0*} \leq z^{1*} \leq \cdots \leq z^{p*} \leq \cdots \leq z^{P*} = z^*, \tag{5.6}$$

where z^{0*} corresponds to the *wait-and-see* solution, while z^{P*} corresponds to the optimal stochastic solution.

However, there is no guarantee that this is the best chain of lower bounds for z^*, that is the one with the lowest gap to the optimal solution. As a matter of fact, alternative groupings (see for example, figures 5.2, 5.3, and 5.4) than the one presented in figure 5.1 could lead to better chains of lower bounds.

For the generic $p \in \mathcal{P}$ level in (5.6), the nonanticipative relaxation as in (5.4) is:

$$z^{p*}(\lambda) = \sum_{m \in \mathcal{M}} \pi^m z^{pm*}(\lambda) \qquad \forall p \in \mathcal{P}, \tag{5.7}$$

where $\quad z^{p,m*}(\lambda) = \min_x \{\sum_{k \in \mathcal{K}} (f_k(x) + \lambda^T A_k x) : x \in \mathcal{X}\},\ \forall m \in \mathcal{M}^p, \forall p \in \mathcal{P}.$

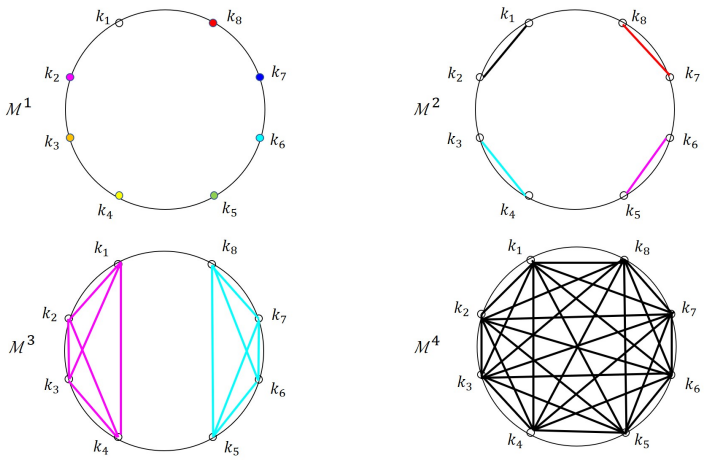


Figure 5.1: A refinement chain with disjoint groups

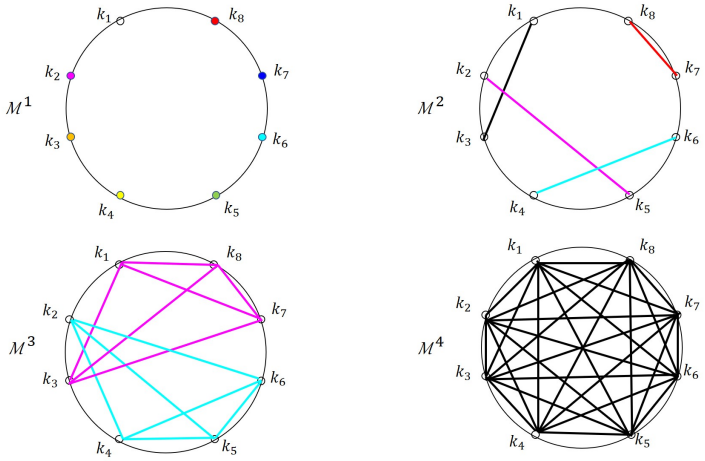


Figure 5.2: An alternative refinement chain with disjoint groups (1)

Notice that the matrices $A_1, \ldots, A_k$ are such that $\sum_{k\in\mathcal{K}} A_k x_k = 0$ if and only if $x_1 = \cdots = x_K$. Consequently, $\sum_{m\in\mathcal{M}}(\sum_{k\in\mathcal{K}} A_k)x^m = 0$ if and only if $x^1 = \cdots = x^M$. As a result, we have that

$$z^{p*}(\lambda) \geq z^*(\lambda) \qquad \forall p \in \mathcal{P} \setminus \{0\},$$

and we would like to construct a refinement level p such that the bound improvement is large.

To compute the bound improvement, we first need to compute the values $z^{pm}(\lambda)$ for all $m \in \mathcal{M}^p$ and, in order to reduce the number of groupings to be examinated, we can use candidate solutions. These candidate solutions can be represented by the set of scenario optimal solutions or can be generated by using any heuristic.

By adapting the computation presented in Ryan, Ahmed, Dey, and Rajan [82] (2016), we have that:

$$0 \leq z^{p*}(\lambda) - z^*(\lambda) \leq \sum_{m\in\mathcal{M}} \theta^{p,m} \qquad \forall p \in \mathcal{P},$$

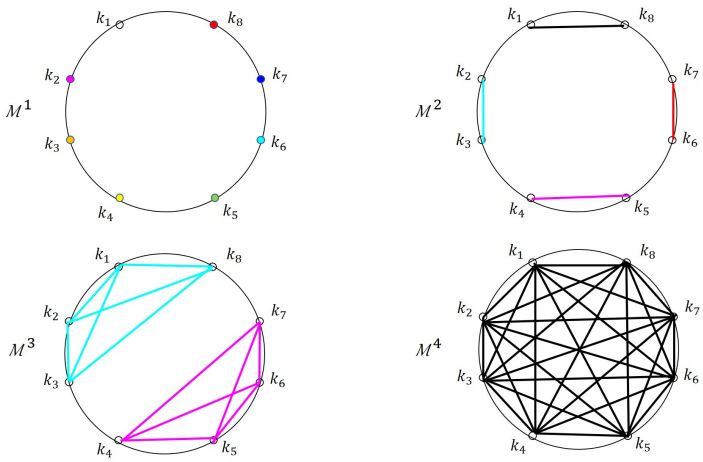


Figure 5.3: An alternative refinement chain with disjoint groups (2)

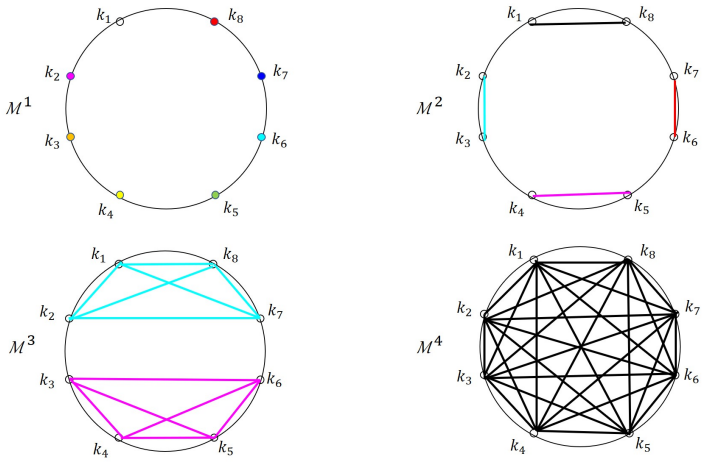


Figure 5.4: An alternative refinement chain with disjoint groups (3)

where:

$$\theta^{p,m} = \min_{x \in \mathcal{S}} \{ \sum_{k \in \mathcal{K}} (f_k(x) + \lambda^T A_k x - z_k^*(\lambda)) \} \qquad \forall p \in \mathcal{P}, m \in \mathcal{M}^p.$$

Hence, $\theta^{p,m}$ is an optimistic prediction of the bound improvement by grouping scenarios.

5.2.2 MIP formulation for scenario grouping

In this section, we propose a mixed integer program whose objective is to maximize the estimated improvement (with respect to the *wait-and-see* solution) which we obtain by grouping scenarios. Instead of the approach of Ryan, Ahmed, Dey, and Rajan [82] (2016), in which they create partitions, by optimizing the scenario grouping refinement levels one by one and by imposing a maximum size to the cardinality of each group, our approach aims to build an optimal refinement chain of lower bounds at

once. This would imply that at some levels of the chain, the improvement is not necessarily the best one. However, we have the guarantee that the overall chain is built such that we obtain the maximum overall estimated improvement considering the construction of the refinement levels. Furthermore, we constrain each group to have a given dimension. On the other hand, our approach differs from the one in Maggioni and Pflug [55] (2016), since in their approach, they build scenario groups in a sequential way, not allowing to guarantee that the chosen grouping provides the best lower bound.

Optimization of refinement chains with disjoint groups

In this section, we first introduce the notation for our problem and secondly, we formulate the scenario grouping problem.

We recall that the set of scenarios is defined by $\mathcal{K} = \{1, \ldots, K\}$, $(k \in \mathcal{K})$, the set of refinement levels by $\mathcal{P} = \{1, \ldots, P\}$, $(p \in \mathcal{P})$, the set of groups in each refinement level p by $\mathcal{M}^p = \{1, \ldots, M^p\}$, $(m \in \mathcal{M}^p)$ and the set of scenario optimal solutions by $\mathcal{S}$, $(x \in \mathcal{S})$. Note that in each refinement level, the following condition $\frac{K}{M} = \mathbb{Z}, \forall p \in \mathcal{P}$ must be satisfied, that is, the number of scenarios in each group for each partition must be an integer number. We introduce the parameter w_{ks} to represent the potential improvement which is computed as the difference between the value of the objective function of the problem where we fix the first-stage variables to be equal to a scenario solution and the solution obtained by solving the problem on that scenario (without fixing any variable). The formula is:

$$w_{ks} = f_k(x) - z_k \qquad \forall k \in \mathcal{K}, \forall x \in \mathcal{S}.$$

Furthermore, we introduce the binary variable y_k^{pm} to indicate whether scenario k belongs to group m in refinement level p and the binary variable v_{ij}^{pm} to indicate whether scenarios i and j belong to the same group m in refinement level p. Finally, the estimated improvement due to group m in refinement level p is represented by the continuous variable θ^{pm}. The adopted notation is summarized as follows:

- Sets:
  - $\mathcal{K}$: set of scenarios, $k \in \mathcal{K}$;
  - $\mathcal{P}$: set of refinement levels, $p \in \mathcal{P}$;
  - $\mathcal{M}^p$: set of groups in each refinement level p, $m \in \mathcal{M}^p$;
  - $\mathcal{S}$: set of optimal solutions, $x \in \mathcal{S}$.
- Variables:
  - y_k^{pm}: 0-1 variable indicating whether we place scenario k into group m in refinement level p;

- v_{ij}^{pm}: 0-1 variable indicating whether we place scenarios i and j in the same group m in refinement level p;
- θ^{pm}: continuous variable representing estimated improvement due to group m in refinement level p.

- Parameters:
  - w_{ks}: potential improvement values for each couple $(k, s), k \in \mathcal{K}, x \in \mathcal{S}$.

Given a finite set of solutions $\mathcal{S}$, values w_{ks}, and a scenario group size $\frac{K}{M}$ for each refinement level $p \in \mathcal{P}$, a mixed integer programming formulation for the scenario disjoint grouping problem is as follows:

$$V(\mathcal{S}, \mathcal{P}, \mathcal{M}^p) = \max_{\theta, y} \sum_{p \in \mathcal{P}} \sum_{m \in M} \theta^{pm} \tag{5.8}$$

s.t.

$$\theta^{pm} \leq \sum_{k \in \mathcal{K}} w_{ks} y_k^{pm} \qquad \forall x \in \mathcal{S}, \forall m \in \mathcal{M}^p, \forall p \in \mathcal{P} \tag{5.9}$$

$$\sum_{m \in \mathcal{M}} y_k^{pm} = 1 \qquad \forall k \in \mathcal{K}, \forall p \in \mathcal{P} \tag{5.10}$$

$$\sum_{k \in \mathcal{K}} y_k^{pm} = \frac{K}{M^p} \qquad \forall p \in \mathcal{P}, \forall m \in \mathcal{M}^p \tag{5.11}$$

$$v_{ij}^{pm} \geq (y_i^{pm} + y_j^{pm}) - 1 \qquad \forall i \in \mathcal{K}, \forall j \in \mathcal{K} : j \neq i, \forall p \in \mathcal{P}, \forall m \in \mathcal{M}^p \tag{5.12}$$

$$2v_{ij}^{pm} \leq y_i^{pm} + y_j^{pm} \qquad \forall i \in \mathcal{K}, \forall j \in \mathcal{K} : j \neq i, \forall p \in \mathcal{P}, \forall m \in \mathcal{M}^p \tag{5.13}$$

$$\sum_{m \in \mathcal{M}} v_{ij}^{pm} \geq \sum_{m \in \mathcal{M}} v_{ij}^{(p-1),m} \qquad \forall i \in \mathcal{K}, \forall j \in \mathcal{K} : j \neq i, \forall p \in \mathcal{P} : p > 1 \tag{5.14}$$

$$y_k^{pm} \in \{0, 1\} \qquad \forall k \in \mathcal{K},\ \forall m \in \mathcal{M}^p,\ \forall p \in \mathcal{P} \tag{5.15}$$

$$v_{ij}^{pm} \in \{0, 1\} \qquad \forall m \in \mathcal{M}^p,\ \forall i, j \in \mathcal{K},\ \forall p \in \mathcal{P} \tag{5.16}$$

$$\theta^{pm} \in \mathbb{R} \qquad \forall m \in \mathcal{M}^p,\ \forall p \in \mathcal{P}. \tag{5.17}$$

The objective function (5.8) maximizes the estimated improvement over all groups and refinement levels. Constraints (5.9) define the improvement associated to a given solution to be the minimum of affine functions and the improvement is considered only if scenario k belongs to that group m in that refinement level p. Constraints (5.10) impose each scenario to belong exactly to one group in each refinement level, while constraints (5.11) assure that the number of scenarios in every group is given by the number of scenarios divided by the number of groups in that refinement level. Furthermore, constraints (5.12) and (5.13) together establish that if both scenarios belong to the same group, v_{ij}^{pm} is equal 1, otherwise equal 0 in each refinement level. Note that equivalently, we can think of two

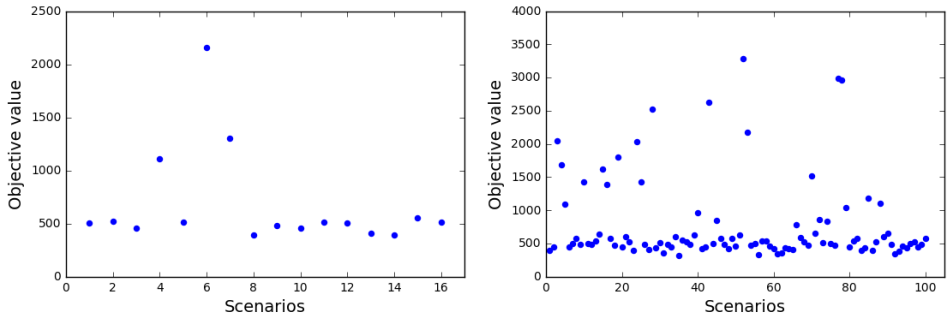
scenarios belonging to the same group as nodes linked by the same arc. Constraints (5.14) guarantee that if in the previous refinement level, scenarios i and j are in the same group, they are in the same group also in this refinement level. Finally, constraints (5.15)-(5.17) are variable definition constraints.

5.3 Computational Results

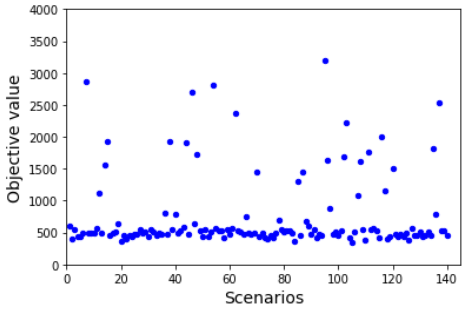
In this section, we present the computational results obtained by applying our methodology.
We first present the results obtained by applying our approach to an instance of the two-stage optimization integer model for the Bike Sharing Allocation and Rebalancing Problem (Cavagnini, Bertazzi, Maggioni, and Hewitt [19] (2018), Chapter 3). Then, we report the results obtained by considering one instance of the Stochastic Server Location Problem (SSLP), available at the SIPLIB (Ahmed, Garcia, Kong, Ntaimo, Parija, Qiu, and Sen [3] (2015)). For both cases, we compare the results obtained by our method, with two alternative approaches. The first one, which we call "Maggioni and Pflug [55] (2016)" refers to the way of grouping scenarios sequentially. The second one, which we call "Ryan, Ahmed, Dey, and Rajan [82] (2016)" refers to the way of grouping scenarios indipendently in each refinement level, but, differently from their approach, we require that each group must have a predefined number of scenarios $\frac{K}{M}$.
All computational experiments were run on a 64-bit machine with 8 GB of RAM and an Intel Core i7-7500 CPU 2.90 GHz processor.

5.3.1 Two-stage stochastic optimization integer model for the Bike sharing allocation and rebalancing problem

In this subsection, we test our approach on the two-stage stochastic optimization model for the Bike sharing allocation and rebalancing problem, by generating different number of scenarios (16, 100, 140). The complete description and formulation of this problem can be found in Chapter 3.
In order to assess if, for this case study, there is a large gap between the wait-and-see problem and the recourse problem, in figure 5.5, we draw the optimal objective function values obtained for each scenario and for every considered instance. We can see that the scenarios show very different objective function optimal values and, consequently, going from one refinement level to the next one, we expect to obtain a large improvement in the lower bounds.
After building scenario groups through our methodology (model (5.8)-(5.17)), we solve the Bikesharing Allocation and Rebalancing Problem on these groups, thus obtaining the chains of lower bounds presented in tables (5.1), (5.2) and (5.3), respectively in cases with 16, 100, 140 scenarios. We recall that, for all cases, we compare the lower bounds z^{*p} obtained by our model (5.8)-(5.17), with the lower



(a) 16 scenarios **(b) 100 scenarios**



(c) 140 scenarios

Figure 5.5: Scenario objective function values for the Bikesharing Allocation and Rebalancing Problem for an increasing number of scenarios.

bounds z^{*p}_{MP} (from Maggioni and Pflug [55] (2016)) and the lower bounds z^{*p}_{RA} (from Ryan, Ahmed, Dey, and Rajan [82] (2016)), with the following formulae: $\frac{z-z}{z}$ and $\frac{z-z}{z}$, respectively.

Across all instances, and across all three grouping approaches (model (5.8)-(5.17), Maggioni and Pflug [55] (2016) and Ryan, Ahmed, Dey, and Rajan [82] (2016)), we can see that the lower bounds are always monotonically increasing.

Concerning the case with 16 scenarios (see table (5.1)), we considered three refinement levels (wait-and-see and complete recourse problem excluded). By comparing our approach with the one of Maggioni and Pflug [55] (2016), we can see that through our method, we always obtain better lower bounds. Furthermore, we notice that even if the estimated improvement given by the approach of Ryan, Ahmed, Dey, and Rajan [82] (2016) is higher than the one of model (5.8)-(5.17), our bounds are overall better, since they are closer to the solution of the full recourse problem.

For the case with 100 scenarios (see table (5.2)), we considered two refinement levels (wait-and-see and complete recourse problem excluded). Also in this case, we notice that our lower bounds are always better than the ones given by the approach of Maggioni and Pflug [55] (2016). However, even if the

estimated improvement of model (5.8)-(5.17) and of the approach of Ryan, Ahmed, Dey, and Rajan [82] (2016) are the same, for the refinement level characterized by four groups made up of 25 scenarios each, Ryan, Ahmed, Dey, and Rajan [82] (2016) approach overperforms the one of model (5.8)-(5.17). Instead, through model (5.8)-(5.17), the best higher lower bound (the one associated to the refinement level composed of 2 groups of 50 scenarios each) is obtained.

Finally, for the case with 140 scenarios (see table (5.3)), we considered two refinement levels (wait-and-see and complete recourse problem excluded). Again, we notice that our lower bounds are always better than the ones given by the approach of Maggioni and Pflug [55] (2016). However, even if the estimated improvement of model (5.8)-(5.17) and of the approach of Ryan, Ahmed, Dey, and Rajan [82] (2016) are the same, for the refinement level characterized by four groups made up of 35 scenarios each, our model (5.8)-(5.17) returns the best lower bound, while the Ryan, Ahmed, Dey, and Rajan [82] (2016) returns the best lower bound in the refinement level with 2 groups composed by 70 scenarios.

			Model (5.8) - (5.17)			Maggioni and Pflug (2016)		Ryan et al (2016)			CPU s/ subproblem	Comparison	
p	M^p	K/M^p	Obj value	% gap wrt z*	Estimated improvement	Obj value	% gap wrt z*	Obj value	% gap wrt z*	Estimated improvement		Maggioni and Pflug (2016) vs Model (5.8) - (5.17)	Ryan et al. (2016) vs Model (5.8)-(5.17)
0 (z_k)	16	1	672.12	8.25%	-	672.12	8.25%	672.12	8.25%	-	0.04	-	-
1	8	2	692.88	5.00%	730.54	691.07	5.28%	690.82	5.32%	768.05	0.05	-0.26%	-0.30%
2	4	4	712.10	2.17%	1189.36	706.77	2.94%	711.66	2.23%	1189.36	0.09	-0.75%	-0.06%
3	2	8	721.26	1.15%	1279.49	716.99	1.47%	713.74	1.93%	1279.49	0.14	-0.59%	-1.04%
4 (z^*)	1	16	727.54	-	-	727.54	-	727.54	-	-	0.39	-	-

Table 5.1: Comparison of lower bounds found through model (5.8)-(5.17), Maggioni and Pflug [55] (2016) and Ryan, Ahmed, Dey, and Rajan [82] (2016) for the Bikesharing problem with 16 scenarios.

			Model (5.8) - (5.17)			Maggioni and Pflug (2016)		Ryan et al (2016)			CPU s/ subproblem	Comparison	
p	M^p	K/M^p	Obj value	% gap wrt z*	Estimated improvement	Obj value	% gap wrt z*	Obj value	% gap wrt z*	Estimated improvement		Maggioni and Pflug (2016) vs Model (5.8) - (5.17)	Ryan et al. (2016) vs Model (5.8)-(5.17)
0 (z_k)	100	1	769.46	13.65%	-	769.46	13.65%	769.46	13.65%	-	0.04	-	-
1	4	25	864.98	1.10%	11095.25	858.18	1.90%	865.95	0.99%	11095.25	0.46	-0.79%	0.11%
2	2	50	870.04	0.51%	11095.25	869.76	0.54%	869.70	0.55%	11095.25	1.01	-0.03%	-0.04%
3 (z^*)	1	100	874.49	-	-	874.49	-	874.49	-	-	2.44	-	-

Table 5.2: Comparison of lower bounds found through model (5.8)-(5.17), Maggioni and Pflug [55] (2016) and Ryan, Ahmed, Dey, and Rajan [82] (2016) for the Bikesharing problem with 100 scenarios.

			Model (5.8) - (5.17)			Maggioni and Pflug (2016)		Ryan et al (2016)			CPU s/ subproblem	Comparison	
p	M^p	K/M^p	Obj value	% gap wrt z*	Estimated improvement	Obj value	% gap wrt z*	Obj value	% gap wrt z*	Estimated improvement		Maggioni and Pflug (2016) vs Model (5.8) - (5.17)	Ryan et al. (2016) vs Model (5.8)-(5.17)
0 (z_k)	140	1	752.32	12.27%	-	752.32	12.27%	752.32	12.27%	-	0.04	-	-
1	4	35	840.91	0.44%	14376.87	837.15	0.89%	839.78	0.57%	14376.87	0.66	-0.45%	-0.13%
2	2	70	843.36	0.15%	14376.87	842.38	0.26%	843.75	0.10%	14376.87	1.45	-0.12%	+0.05%
3 (z^*)	1	140	844.60	-	-	844.60	-	844.60	-	-	3.56	-	-

Table 5.3: Comparison of lower bounds found through model (5.8)-(5.17), Maggioni and Pflug [55] (2016) and Ryan, Ahmed, Dey, and Rajan [82] (2016) for the Bikesharing problem with 140 scenarios.

5.3.2 SSLP 10 50 50

In this subsection, we test our approach by considering an instance of the Stochastic Server Location Problem (available at the SIPLIB (Ahmed, Garcia, Kong, Ntaimo, Parija, Qiu, and Sen [3] (2015))), for the case with 10 first stage variables, 50 second-stage variables per scenario and 50 scenarios.
As for the problem analyzed in the previous subsection, we first represent the objective function values for each scenario (figure 5.6). From the graph, we notice that the objective function values vary from scenario to scenario, and, consequently, we expect to obtain better and better lower bounds going from one refinement level to the next one.

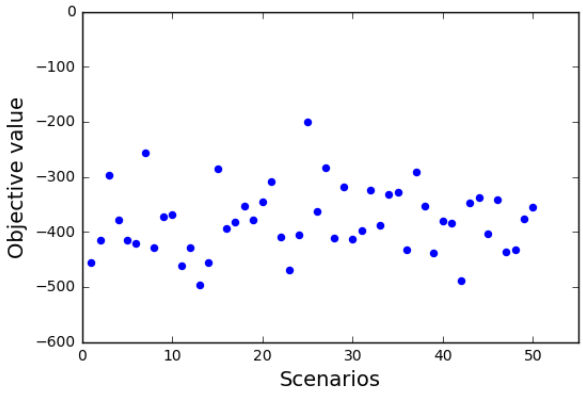


Figure 5.6: Scenario objective function values for the $SSLP\_10\_50\_50$

After obtaining scenario groups by applying our methodology (model (5.8)-(5.17)), we solve the SSLP on these groups, thus obtaining the chain of lower bounds displayed in table (5.4). Also in this case, we compare the lower bounds z^{*p}_{MP} (from Maggioni and Pflug [55] (2016)) and the lower bounds z^{*p}_{RA} (from Ryan, Ahmed, Dey, and Rajan [82] (2016)), with the following formulae: $\frac{z-z}{z}$ and $\frac{z-z}{z}$, respectively. We specify that the optimal value z^* of this instance of the SSLP has been taken from Ryan, Ahmed, Dey, and Rajan [82] (2016), as with our machine we were not able to get the optimal solution after one hour. The reader may refer to Ryan, Ahmed, Dey, and Rajan [82] (2016) for the summary statistics for this problem.
For this instance, we considered two refinement levels (wait-and-see and complete recourse problem excluded). We notice that the lower bounds obtained by model (5.8)-(5.17) are always better than the ones computed according to Maggioni and Pflug [55] (2016), showing the validity of our approach. Furthermore, in this case, our lower bounds are equal to the ones obtained by optimizing the groups construction separately for each refinement level as done in Ryan, Ahmed, Dey, and Rajan [82] (2016) and, thus, by solving two optimization problems, instead of one (model (5.8)-(5.17)).

			Model (5.8) - (5.17)			Maggioni and Pflug (2016)		Ryan et al. (2016)			CPU s subproblem	Comparisons	
p	M^p	K/M^p	Obj value	% gap wrt z*	Estimated improvement	Obj value	% gap wrt z*	Obj value	% gap wrt z*	Estimated improvement		Maggioni and Pflug (2016) vs Model (5.8) - (5.17)	Ryan et al. (2016) vs Model (5.8)-(5.17)
0 (z_k)	50	1	-378.92	3.92%	-	-378.92	3.92%	-378.92	3.92%	-	0.19	-	-
1	10	5	-364.64	0.00%	714	-366.46	0.50%	-364.64	0.00%	714	2.80	0.50%	0.00%
2	5	10	-364.64	0.00%	714	-364.86	0.06%	-364.64	0.00%	714	5.19	0.06%	0.00%
3 (z^*)	1	50	-364.64	-	-	-364.64	-	-364.64	-	-	*	-	-

Table 5.4: Comparison of lower bounds for the $SSLP\_10\_50\_50$, found through model (5.8)-(5.17), Maggioni and Pflug [55] (2016) and Ryan, Ahmed, Dey, and Rajan [82] (2016).

5.4 Conclusions

In this chapter, we have proposed a mixed integer linear program in order to obtain an optimized chain of lower bounds for two-stage stochastic programs by solving subproblems based on scenario groupings. Differently from the approaches proposed in the literature (see Ryan, Ahmed, Dey, and Rajan [82] (2016) and Maggioni and Pflug [55] (2016)), we build disjoint scenario groups with the objective of maximizing the estimated improvement obtained by solving the stochastic program on scenario subgroups with respect to the wait-and-see solution. Through the computational experiments, we showed that our methodology always improves the chain of lower bounds built according to the approach proposed by Maggioni and Pflug [55] (2016). Compared to the approach of Ryan, Ahmed, Dey, and Rajan [82] (2016), we provided a monotonic chain of lower bounds, instead of a single lower bound, and, in some cases, our methodology outperforms their approach.

Given that the computational complexity of the grouping model (5.8)-(5.17) grows both with the number of scenarios and with the number of groups, as future developments, we aim at investigating other methods to overcome the computational difficulties encountered in this work.

Appendices

Appendix 1

1.a Model linearization

In this section, we present the linearization of Problem $\mathcal{P}1$. The model is now:

$$\text{Problem } \mathcal{P}1^L$$

$$\begin{aligned}\min \sum_{t\in\mathcal{T}}\sum_{n\in\mathcal{N}} \pi^n \Big[\sum_{i\in\mathcal{I}}(f_i x_i^n + F_i v_i^n)\Big] + \\ + \sum_{t\in\mathcal{T}}\sum_{n\in\mathcal{N}} \pi^n \Big[\sum_{i\in\mathcal{I}}(g_i b_i^n + G_i r_i^n) + \sum_{i\in\mathcal{I}} h_i I_i^{n+} + \\ + \sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{I}:i\neq j}(t_{ij} y_{ij}^{n+} + T_{ij} V_{ij}^n) + \sum_{j\in\mathcal{I}} p_j I_j^{n-}\Big]\end{aligned} \tag{1.a.1}$$

s.t. (2.4), (2.5), (2.6), (2.8)-(2.14), and the following additional constraints:

$$I_i^n = I_i^{a(n)+} + x_i^{a(n)} + b_i^n - d_i^n + \sum_{j\in\mathcal{I}:i\neq j}(y_{ji}^n - y_{ij}^n) \qquad i\in\mathcal{I}, n\in\mathcal{N}^t,\ t\in\mathcal{T}'' \tag{1.a.2}$$

$$I_i^{a(n)+} + x_i^{a(n)} + b_i^n + \sum_{j\in\mathcal{I}:j\neq i} y_{ji}^n \leq Q_i \qquad i\in\mathcal{I}, n\in\mathcal{N}^t,\ t\in\mathcal{T}'' \tag{1.a.3}$$

$$\sum_{j\in\mathcal{I}:j\neq i} y_{ij}^n \leq \gamma_i^n \qquad i\in\mathcal{I}, n\in\mathcal{N}^t,\ t\in\mathcal{T}'' \tag{1.a.4}$$

$$\gamma_i^n \geq (I_i^{a(n)+} + x_i^{a(n)} - d_i^n) \qquad i\in\mathcal{I}, n\in\mathcal{N}^t,\ t\in\mathcal{T}'' \tag{1.a.5}$$

$$\gamma_i^n \leq (I_i^{a(n)+} + x_i^{a(n)} - d_i^n) l_i^n \qquad i\in\mathcal{I}, n\in\mathcal{N}^t,\ t\in\mathcal{T}'' \tag{1.a.6}$$

$$\gamma_i^n \leq (I_i^{a(n)+} + x_i^{a(n)} - d_i^n) + d_i^n(1-l_i^n) \qquad i\in\mathcal{I}, n\in\mathcal{N}^t,\ t\in\mathcal{T}'' \tag{1.a.7}$$

$$I_i^{n+} \geq I_i^n \qquad i\in\mathcal{I},\ n\in\mathcal{N}^t,\ t\in\mathcal{T}'' \tag{1.a.8}$$

$$I_i^{n+} \leq I_i^n + M(1-z_i^n) \qquad i\in\mathcal{I},\ n\in\mathcal{N}^t,\ t\in\mathcal{T}'' \tag{1.a.9}$$

$$I_i^{n+} \geq 0\ integer \qquad i\in\mathcal{I},\ n\in\mathcal{N}^t,\ t\in\mathcal{T}'' \tag{1.a.10}$$

$$I_i^{n+} \leq 0 + M z_i^n \qquad i\in\mathcal{I},\ n\in\mathcal{N}^t,\ t\in\mathcal{T}'' \tag{1.a.11}$$

$$I_i^{n-} \geq -I_i^n \qquad i\in\mathcal{I},\ n\in\mathcal{N}^t,\ t\in\mathcal{T}'' \tag{1.a.12}$$

$$I_i^{n-} \leq -I_i^n + M z_i^n \qquad i\in\mathcal{I},\ n\in\mathcal{N}^t,\ t\in\mathcal{T}'' \tag{1.a.13}$$

$$I_i^{n-} \geq 0\ integer \qquad i\in\mathcal{I},\ n\in\mathcal{N}^t,\ t\in\mathcal{T}'' \tag{1.a.14}$$

$$I_i^{n-} \leq 0 + M(1-z_i^n) \qquad i\in\mathcal{I},\ n\in\mathcal{N}^t,\ t\in\mathcal{T}'' \tag{1.a.15}$$

$$\gamma_i^n \geq\ integer \qquad i\in\mathcal{I}, n\in\mathcal{N}^t,\ t\in\mathcal{T}'' \tag{1.a.16}$$

$$l_i^n \in \{0,1\} \qquad i\in\mathcal{I}, n\in\mathcal{N}^t,\ t\in\mathcal{T}'' \tag{1.a.17}$$

$$z_i^n \in \{0,1\} \qquad i\in\mathcal{I},\ n\in\mathcal{N}^t,\ t\in\mathcal{T}'' \tag{1.a.18}$$

where $M = \max\{Q_i, max_{n\in\mathcal{N}} d_i^n\}$.

Appendix 2

2.a Model linearization

In this section, we present the linearization of the model presented in section 3.4. First, we need to modify the objective function (3.1) as follows:

$$min \sum_{i\in\mathcal{I}} f_i x_i + \sum_{s\in\mathcal{S}} pr^s [\sum_{i\in\mathcal{I}\setminus\{I\}} (t_{i,i+1} y^s_{i,i+1} + \frac{c_i}{Q_i} B_i^{s+} + c_i E_i^{s+} + p_i I_i^{s-})]. \tag{2.a.1}$$

Then, we need to linearize the expressions for determining I_i^{s+} and I_i^{s-}. We introduce the binary variable z_i^s such that:

$$z_i^s = \begin{cases} 1 & \text{if } I_i^s \geq 0 \\ 0 & \text{otherwise} \end{cases}$$

and we substitute constraints (3.10) and (3.11) with:

$$I_i^{s+} \geq I_i^s \qquad i \in \mathcal{I}\setminus\{I\},\ s\in\mathcal{S} \tag{2.a.2}$$

$$I_i^{s+} \leq I_i^s + \max_{s\in\mathcal{S}}(d_i^s)(1-z_i^s) \qquad i \in \mathcal{I}\setminus\{I\},\ s\in\mathcal{S} \tag{2.a.3}$$

$$I_i^{s+} \geq 0 \ \ integer \qquad i \in \mathcal{I}\setminus\{I\},\ s\in\mathcal{S} \tag{2.a.4}$$

$$I_i^{s+} \leq M z_i^s \qquad i \in \mathcal{I}\setminus\{I\},\ s\in\mathcal{S} \tag{2.a.5}$$

$$I_i^{s-} \geq -I_i^s \qquad i \in \mathcal{I}\setminus\{I\},\ s\in\mathcal{S} \tag{2.a.6}$$

$$I_i^{s-} \leq -I_i^s + M z_i^s \qquad i \in \mathcal{I}\setminus\{I\},\ s\in\mathcal{S} \tag{2.a.7}$$

$$I_i^{s-} \geq 0 \ \ integer \qquad i \in \mathcal{I}\setminus\{I\},\ s\in\mathcal{S} \tag{2.a.8}$$

$$I_i^{s-} \leq \max_{s\in\mathcal{S}}(d_i^s)(1-z_i^s) \qquad i \in \mathcal{I}\setminus\{I\},\ s\in\mathcal{S} \tag{2.a.9}$$

The same linearization technique is applied for E_i^{s+}, and B_i^{s+}. In particular, we introduce the binary variables e_i^s and r_i^s, such that:

$$e_i^s = \begin{cases} 1 & \text{if } E_i^s \geq 0 \\ 0 & \text{otherwise} \end{cases}$$

$$r_i^s = \begin{cases} 1 & \text{if } B_i^s \geq 0 \\ 0 & \text{otherwise} \end{cases}$$

and we substitute constraints (3.13) and (3.15) with the following:

$$B_i^{s+} \geq B_i^s \qquad i \in \mathcal{I}\setminus\{I\},\ s\in\mathcal{S} \tag{2.a.10}$$

$$B_i^{s+} \leq B_i^s + M(1-r_i^s) \qquad i \in \mathcal{I}\setminus\{I\},\ s\in\mathcal{S} \tag{2.a.11}$$

$$B_i^{s+} \leq Mr_i^s \qquad i \in \mathcal{I} \setminus \{I\},\ s \in \mathcal{S} \tag{2.a.12}$$

$$B_i^{s+} \geq 0 \ \ integer \qquad i \in \mathcal{I} \setminus \{I\},\ s \in \mathcal{S} \tag{2.a.13}$$

$$B_i^{s-} \geq -B_i^s \qquad i \in \mathcal{I} \setminus \{I\},\ s \in \mathcal{S} \tag{2.a.14}$$

$$B_i^{s-} \leq -B_i^s + Mr_i^s \qquad i \in \mathcal{I} \setminus \{I\},\ s \in \mathcal{S} \tag{2.a.15}$$

$$B_i^{s-} \leq M(1 - r_i^s) \qquad i \in \mathcal{I} \setminus \{I\},\ s \in \mathcal{S} \tag{2.a.16}$$

$$B_i^{s-} \geq 0 \ \ integer \qquad i \in \mathcal{I} \setminus \{I\},\ s \in \mathcal{S} \tag{2.a.17}$$

$$E_i^{s+} \geq E_i^s \qquad i \in \mathcal{I} \setminus \{I\},\ s \in \mathcal{S} \tag{2.a.18}$$

$$E_i^{s+} \leq E_i^s + M(1 - e_i^s) \qquad i \in \mathcal{I} \setminus \{I\},\ s \in \mathcal{S} \tag{2.a.19}$$

$$E_i^{s+} \leq Me_i^s \qquad i \in \mathcal{I} \setminus \{I\},\ s \in \mathcal{S} \tag{2.a.20}$$

$$E_i^{s+} \geq 0 \ \ integer \qquad i \in \mathcal{I} \setminus \{I\},\ s \in \mathcal{S} \tag{2.a.21}$$

$$E_i^{s-} \geq -E_i^s \qquad i \in \mathcal{I} \setminus \{I\},\ s \in \mathcal{S} \tag{2.a.22}$$

$$E_i^{s-} \leq -E_i^s + Me_i^s \qquad i \in \mathcal{I} \setminus \{I\},\ s \in \mathcal{S} \tag{2.a.23}$$

$$E_i^{s-} \leq M(1 - e_i^s) \qquad i \in \mathcal{I} \setminus \{I\},\ s \in \mathcal{S} \tag{2.a.24}$$

$$E_i^{s-} \geq 0 \ \ integer \qquad i \in \mathcal{I} \setminus \{I\},\ s \in \mathcal{S} \tag{2.a.25}$$

Finally, we introduce the variable definition constraints:

$$z_i^s \in \{0, 1\} \qquad i \in \mathcal{I} \setminus \{I\},\ s \in \mathcal{S} \tag{2.a.26}$$

$$r_i^s \in \{0, 1\} \qquad i \in \mathcal{I} \setminus \{I\},\ s \in \mathcal{S} \tag{2.a.27}$$

$$e_i^s \in \{0, 1\} \qquad i \in \mathcal{I} \setminus \{I\},\ s \in \mathcal{S} \tag{2.a.28}$$

2.b Determining the initial bike requirement for each station

To ensure each station has a sufficient number of bikes to satisfy rental requests early in the day, before any bikes are returned to that station, we determine an initial bike requirement for each station. We consider five methods for estimating the number of withdrawn bikes before a return occurs, and test their relative performance with our simulation. All are based on calculating statistics from historical ridership data over the time period from 6 am until 11:59 am. These methods are based on one of the following statistics for each station: (1) Average bike-interarrival time for bikes returned to that station, (2) The average trip time for bikes returned to that station, (3) The average time until the first return to that station, and, (4) The total number of bikes withdrawn from that station between 6 am and 11:59 am. The methods are as follows:

- Method 1: We estimate the initial requirement as the average number of withdrawn bikes between 6 a.m. and 6 a.m., plus the average bike inter-arrival time.
- Method 2: We estimate the initial requirement as the average number of withdrawn bikes between 6 a.m. and 6 a.m., plus thie average trip time.
- Method 3: We estimate the initial requirement as this total number of withdrawn bikes divided by the average bike inter-arrival time.
- Method 4: We estimate the initial requirement as this total number of withdrawn bikes divided by the average trip time.
- Method 5: We estimate the initial requirement as the total number of withdrawn bikes between 6 a.m. and 6 a.m., plus the average time until first return.

To assess each method, we first solve the stochastic program, while requiring that the number of bikes initially allocated to each station is at least as great as the number suggested by that method. We then run our simulation, with those initial allocations, and compute the frequency of congestion (starvation) relative to the number of bikes returned (withdrawn). We report these relative frequencies in Table 2.b.1. There, we see that methods 3 and 5 perform the best with respect to starvation, with 5 performing slightly better with respect to congestion. We also see that fewer bikes are allocated with method 5. Thus, we conclude that method 5 is the best method, and used it for the remainder of our computational study.

	Method 1		Method 2		Method 3		Method 4		Method 5	
	% Congestion	% Starvation	% Congestion	% Starvation	% Congestion	% Starvation	% Congestion	% Starvation	% Congestion	% Starvation
Mon	8.43%	33.07%	8.43%	33.07%	9.00%	29.53%	8.43%	32.68%	8.81%	31.30%
Tue	8.64%	30.94%	8.64%	31.14%	9.41%	28.74%	8.64%	30.54%	9.21%	28.74%
Wed	7.07%	32.57%	7.07%	32.57%	7.07%	30.48%	7.07%	32.57%	7.27%	31.11%
Thu	6.94%	30.19%	6.94%	30.19%	7.14%	28.30%	6.94%	30.19%	6.94%	27.88%
Fri	6.68%	28.97%	6.68%	28.97%	6.91%	25.29%	6.68%	28.97%	6.68%	25.98%
Sat	7.62%	32.04%	7.62%	33.01%	7.62%	17.48%	7.62%	30.10%	7.62%	21.36%
Sun	0.00%	29.17%	0.00%	31.25%	0.00%	20.83%	0.00%	31.25%	0.00%	20.83%
Average	**6.48%**	**30.99%**	**6.48%**	**31.46%**	**6.74%**	**25.81%**	**6.48%**	**30.90%**	**6.65%**	**26.74%**
Initial requirement	55		36		103		57		89	
Total delivery	147		146		160		148		154	

Table 2.b.1: Congestion and starvation relative frequencies by method of determining initial bike requirement

2.c The Sequence based Heuristic (SBH)

In this section, we present the SBH algorithm in detail (see Algorithm (2)).

Lines 1-17 represent the situation in which the sum of the maximum quantity between the one suggested by the newsvendor approach and the initial requirements is less than the initial availability at the depot; in this case, if the quantity to be delivered to a station is less or equal than the station capacity, we consider x_i to be equal to the amount suggested by the newsvendor approach or the initial requirement (lines 3-4). Otherwise, we fix x_i equal to the station capacity and the balance quantity is delivered to a station that occurs earlier in the route and has space left for additional bikes (lines 6-17). On the other hand, lines 18-45 describe the situation in which the sum of the maximum quantity between the one suggested by the newsvendor approach and the initial requirements is greater than the initial availability at the depot; firstly, the initial requirements quantities are assigned to stations (lines 19-21); then, if the quantity suggested by the newsvendor approach is greater than the initial required quantity, the minimum between the quantity suggested by the newsvendor approach and the initial availability at the depot is delivered, starting from the first visited stations (lines 22-25); moreover, if the capacity is sufficient, the delivered quantity is equal to that amount (lines 26-28); otherwise, the balance is delivered to the most preceeding station with left capacity (lines 30-43).

Algorithm 2: Sequence-based heuristic (SBH)

Input: the first-stage decision variables $x_i, i \in \mathcal{I} \setminus \{I\}$ established through the newsvendor approach, Q, $\overline{I}$, $\underline{x}$;

Output: $x_i, i \in \mathcal{I} \setminus \{I\}$ to use for the second-stage program;

if $\sum max(x_i, \underline{x}_i) \leq \overline{I}$ **then**
- **for** $i \in \mathcal{I}$ **do**
  - **if** $max(x_i, \underline{x}_i) \leq Q - \overline{I}$ **then**
    - $x_i = max(x_i, \underline{x}_i)$
  - **else**
    - $x_i = Q - \overline{I}$
    - $balance = max(x_i, \underline{x}_i) - x_i$
    - $l = 1$
    - **while** $balance > 0$ *and* $i - l > 0$ **do**
      - **if** $Q_{-i} - x_{-i} - \overline{I}_{-i} > 0$ **then**
        - $originalx_i = x_i$
        - $x_i += min(Q_{-i} - x_{-i} - \overline{I}_{-i}, balance)$
        - $increase_i = x_i - originalx_i$
        - $balance = balance - increase_i$
        - $l += 1$
      - **else**
        - $l += 1$

else
- **for** $i \in \mathcal{I}$ **do**
  - $x_i = \underline{x}_i$
  - $\overline{I}_i = \overline{I}_i - x_i$
- **while** $\overline{I}_{-i} > 0$ **do**
  - **for** $i \in \mathcal{I}$ **do**
    - **if** $x_{-i} - x_{-i} > 0$ **then**
      - $x_i = min(\overline{I}_i, x_{-i} - x_{-i})$
      - **if** $x_i \leq Q - x_i$ **then**
        - $x_i = x_i + x_i$
        - $\overline{I}_i = \overline{I}_i - x_i$
      - **else**
        - $x_i = x_i + (Q - x_i)$
        - $\overline{I}_i = \overline{I}_i - (Q - x_i)$
        - $balance = x_i - (Q - x_i)$
        - $l = 1$
        - **while** $balance > 0$ *and* $i - l > 0$ **do**
          - **if** $Q_{-i} - x_{-i} - \overline{I}_{-i} > 0$ **then**
            - $originalx_i = x_i$
            - $x_i += min(Q_{-i} - x_{-i} - \overline{I}_{-i}, balance)$
            - $increase_i = x_i - originalx_i$
            - $balance = balance - increase_i$
            - $\overline{I}_i = \overline{I}_i - increase_i$
            - $l += 1$
          - **else**
            - $l += 1$
    - **else**
      - $x_i = x_i$

Appendix 3

3.a Reformulation

Since the model described in section (4.4.1) is non-linear and the productivity rate function has a finite and discrete domain, we adapt the reformulation technique presented by Hewitt, Chacosky, Grasman, and Thomas [44] (2015) to our case. Due to the stochasticity presence, our reformulated model will have a higher dimension, since now we have an enumeration for each couple worker-production cell, for each period and for each scenario. To clarify this point, the following algorithm can be helpful.

Algorithm 3: Enumeration

```
for s = 1, ..., S do
    for i = 1, ..., I do
        for j = 1, ..., J do
            for t = 1, ..., T do
                r̄^{tts}_{ij} = I_{ij} + K_{ij}[1 − e^{-}]
                for k = 0, ..., t do
                    r̄^{kts}_{ij} = I_{ij} + K_{ij}[1 − e^{-}]
                end
            end
        end
    end
end
```

We indicate by $\overline{r}_{ij}^{kts}$ the enumeration of the productivity rate for each scenario, for each period and for each couple worker-production cell, while $\overline{r}_{ij}^{tts}$ represents the maximum productivity rate in each period for each couple worker-production cell in each scenario, corresponding to the highest achievable productivity in the case in which we associate that worker to that production cell in every period up to the considered period. We apply the reformulation to Å, by introducing the binary variables z_{ij}^{kt} to indicate if the productivity rate $\overline{r}^{k}$ is associated to worker i at task j in period t. Moreover, we substitute constraint sets (4.9) and (4.10) with the following and we call the reformulated problem Problem $\mathcal{A}^L$:

$$o_{ij}^{ts} \leq \overline{r}_{ij}^{tts} x_{ij}^{t} \qquad i \in \mathcal{I}, j \in \mathcal{J}, t \in \mathcal{T}, s \in \mathcal{S} \tag{3.a.1}$$

$$o_{ij}^{ts} \leq r_{ij}^{ts} \qquad i \in \mathcal{I}, j \in \mathcal{J}, t \in \mathcal{T}, s \in \mathcal{S} \tag{3.a.2}$$

$$r_{ij}^{ts} = \sum_{k=0}^{t} \bar{r}_{ij}^{kts} z_{ij}^{kt} \qquad i \in \mathcal{I}, j \in \mathcal{J}, t \in \mathcal{T}, s \in \mathcal{S} \tag{3.a.3}$$

$$\sum_{k=0}^{t} k z_{ij}^{kt} \leq \sum_{k=1}^{t} x_{ij}^{k} \qquad i \in \mathcal{I}, j \in \mathcal{J}, t \in \mathcal{T} \tag{3.a.4}$$

$$\sum_{k=0}^{t} z_{ij}^{kt} = x_{ij}^{t} \qquad i \in \mathcal{I}, j \in \mathcal{J}, t \in \mathcal{T} \tag{3.a.5}$$

$$z_{ij}^{kt} \in \{0,1\} \qquad i \in \mathcal{I}, j \in \mathcal{J}, t \in \mathcal{T}, k \in \mathcal{T} : k \leq t. \tag{3.a.6}$$

Constraints (3.a.1) linearize constraints (4.9), taking into consideration the maximum achievable productivity rate for worker i at production cell j in period t and scenario s. Constraints (3.a.2) represent an upper bound on the output of worker i at production cell j in each period and scenario. Constraints (3.a.3) and (3.a.4) guarantee that the productivity rate of a worker in period t coincides with the number of times that the worker has performed the task in the periods up to and including t. Finally, constraints (3.a.5) make sure that each individual is assigned exactly one rate in each period and constraints (3.a.6) impose that variable z is binary.

3.b Detailed regression results

Dep. Variable:	*na*	**R-squared:**	0.627
Model:	OLS	**Adj. R-squared:**	0.626
Method:	Least Squares	**F-statistic:**	602.5
Date:	Fri, 18 May 2018	**Prob (F-statistic):**	3.33e-305
Time:	10:58:07	**Log-Likelihood:**	-3895.2
No. Observations:	1440	**AIC:**	7800.
Df Residuals:	1435	**BIC:**	7827.
Df Model:	4		

	coef	**std err**	**t**	**P>|t|**	**[0.025**	**0.975]**
β	6.1597	0.096	64.485	0.000	5.972	6.347
μ	-6.0808	0.485	-12.534	0.000	-7.032	-5.129
σ^2	-0.9677	0.245	-3.958	0.000	-1.447	-0.488
$FTAP$	-12.9385	0.626	-20.677	0.000	-14.166	-11.711
TCP	14.3039	0.743	19.261	0.000	12.847	15.761

Omnibus:	39.787	**Durbin-Watson:**	2.266
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	55.536
Skew:	0.289	**Prob(JB):**	8.72e-13
Kurtosis:	3.769	**Cond. No.**	9.65

Table 3.b.1: Detailed linear regression results for na

Dep. Variable:	*np*	**R-squared:**	0.579
Model:	OLS	**Adj. R-squared:**	0.578
Method:	Least Squares	**F-statistic:**	493.6
Date:	Fri, 18 May 2018	**Prob (F-statistic):**	9.43e-268
Time:	10:58:04	**Log-Likelihood:**	-880.36
No. Observations:	1440	**AIC:**	1771.
Df Residuals:	1435	**BIC:**	1797.
Df Model:	4		

	coef	**std err**	**t**	**P>|t|**	**[0.025**	**0.975]**
β	0.6931	0.012	58.874	0.000	0.670	0.716
μ	-0.1340	0.060	-2.242	0.025	-0.251	-0.017
σ^2	-0.3255	0.030	-10.803	0.000	-0.385	-0.266
$FTAP$	-0.5788	0.077	-7.506	0.000	-0.730	-0.428
TCP	1.2190	0.092	13.319	0.000	1.039	1.398

Omnibus:	133.259	**Durbin-Watson:**	2.051
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	169.487
Skew:	0.816	**Prob(JB):**	1.57e-37
Kurtosis:	3.404	**Cond. No.**	9.65

Table 3.b.2: Detailed linear regression results for np

Dep. Variable:	*CTI*	**R-squared:**	0.278
Model:	OLS	**Adj. R-squared:**	0.274
Method:	Least Squares	**F-statistic:**	62.31
Date:	Sat, 19 May 2018	**Prob (F-statistic):**	5.49e-55
Time:	07:45:52	**Log-Likelihood:**	18.220
No. Observations:	814	**AIC:**	-24.44
Df Residuals:	808	**BIC:**	3.771
Df Model:	5		

	coef	**std err**	**t**	**P>|t|**	**[0.025**	**0.975]**
β	1.5345	0.021	74.795	0.000	1.494	1.575
μ	0.0616	0.049	1.250	0.212	-0.035	0.158
σ^2	0.1653	0.030	5.443	0.000	0.106	0.225
$FTAP$	0.0202	0.051	0.392	0.695	-0.081	0.121
TCP	0.0799	0.079	1.013	0.311	-0.075	0.235
CI	-0.0013	9.57e-05	-13.252	0.000	-0.001	-0.001

Omnibus:	62.870	**Durbin-Watson:**	1.838
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	75.127
Skew:	-0.733	**Prob(JB):**	4.86e-17
Kurtosis:	2.743	**Cond. No.**	2.30e+03

Table 3.b.3: Detailed linear regression results for *CTI*

Dep. Variable:	PI	**R-squared:**	0.282
Model:	OLS	**Adj. R-squared:**	0.278
Method:	Least Squares	**F-statistic:**	63.48
Date:	Sat, 19 May 2018	**Prob (F-statistic):**	6.82e-56
Time:	07:46:36	**Log-Likelihood:**	594.22
No. Observations:	814	**AIC:**	-1176.
Df Residuals:	808	**BIC:**	-1148.
Df Model:	5		

	coef	**std err**	**t**	**P>|t|**	**[0.025**	**0.975]**
β	0.4912	0.010	48.578	0.000	0.471	0.511
μ	-0.0266	0.024	-1.096	0.274	-0.074	0.021
σ^2	0.0303	0.015	2.027	0.043	0.001	0.060
$FTAP$	-0.0012	0.025	-0.045	0.964	-0.051	0.049
TCP	-0.1431	0.039	-3.681	0.000	-0.219	-0.067
CI	0.0008	4.71e-05	16.155	0.000	0.001	0.001

Omnibus:	327.982	**Durbin-Watson:**	2.006
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	2708.615
Skew:	-1.604	**Prob(JB):**	0.00
Kurtosis:	11.341	**Cond. No.**	2.30e+03

Table 3.b.4: Detailed linear regression results for PI

References

The references list at pages 129 - 136 has been elaborated under the bibliographic standards of the Italian scientific sector MAT/09 Operations Research.

[1] Saligrama R Agnihothri and Ajay K Mishra. Cross-training decisions in field services with three job types and server–job mismatch. *Decision Sciences*, 35(2):239–257, 2004.

[2] SR Agnihothri, AK Mishra, and DE Simmons. Workforce cross-training decisions in field service systems with two job types. *Journal of the Operational Research Society*, 54(4):410–418, 2003.

[3] Shabbir Ahmed, R Garcia, N Kong, L Ntaimo, G Parija, F Qiu, and S Sen. Siplib: A stochastic integer programming test problem library. *See http://www2. isye. gatech. edu/˜ sahmed/siplib*, 2015.

[4] Michel Jose Anzanello and Flavio Sanson Fogliatto. Learning curve models and applications: Literature review and research directions. *International Journal of Industrial Ergonomics*, 41(5): 573–583, 2011.

[5] Sven Axsäter. Evaluation of unidirectional lateral transshipments and substitutions in inventory systems. *European Journal of Operational Research*, 149(2):438–447, 2003.

[6] Sven Axsäter. A new decision rule for lateral transshipments in inventory systems. *Management Science*, 49(9):1168–1179, 2003.

[7] Sudipto Banerjee. On geodetic distance computations in spatial modeling. *Biometrics*, 61(2): 617–625, 2005.

[8] Mike Benchimol, Pascal Benchimol, Benoît Chappert, Arnaud De La Taille, Fabien Laroche, Frédéric Meunier, and Ludovic Robinet. Balancing the stations of a self service “bike hire” system. *RAIRO-Operations Research*, 45(1):37–61, 2011.

[9] Gerardo Berbeglia, Jean-François Cordeau, Irina Gribkovskaia, and Gilbert Laporte. Static pickup and delivery problems: a classification scheme and survey. *Top*, 15(1):1–31, 2007.

[10] Luca Bertazzi and Francesca Maggioni. Solution approaches for the stochastic capacitated traveling salesmen location problem with recourse. *Journal of Optimization Theory and Applications*, 166(1):321–342, 2015.

[11] Luca Bertazzi and Francesca Maggioni. A stochastic multi-stage fixed charge transportation problem: Worst-case analysis of the rolling horizon approach. *European Journal of Operational Research*, 267(2):555–569, 2018.

[12] John R Birge and Francois Louveaux. *Introduction to stochastic programming*. Springer Science & Business Media, 2011.

[13] Jan Brinkmann, Marlin W Ulmer, and Dirk C Mattfeld. Short-term strategies for stochastic inventory routing in bike sharing systems. *Transportation Research Procedia*, 10:364–373, 2015.

[14] Jan Brinkmann, Marlin W Ulmer, and Dirk C Mattfeld. Inventory routing for bike sharing systems. *Transportation Research Procedia*, 19:316–327, 2016.

[15] Michael J Brusco and Tony R Johns. Staffing a multiskilled workforce with varying levels of productivity: An analysis of cross-training policies. *Decision Sciences*, 29(2):499–515, 1998.

[16] Gérard Cachon and Christian Terwiesch. *Matching supply with demand: An introduction to operations management*. Irwin Professional Pub, 2009.

[17] Gerard M Campbell. Cross-utilization of workers whose capabilities differ. *Management Science*, 45(5):722–732, 1999.

[18] Rossana Cavagnini, Luca Bertazzi, and Francesca Maggioni. A two-stage stochastic model for distribution logistics with transshipment and backordering: stochastic vs deterministic solutions. *International Conference on Optimization and Decision Science, ODS 2018*, AIRO Springer Series, 2018.

[19] Rossana Cavagnini, Luca Bertazzi, Francesca Maggioni, and Mike Hewitt. A two-stage stochastic optimization model for the bike sharing allocation and rebalancing problem. *Submitted for evaluation in Omega*, 2018.

[20] Suresh Chand, Vernon Ning Hsu, and Suresh Sethi. Forecast, solution, and rolling horizons in operations management problems: A classified bibliography. *Manufacturing & Service Operations Management*, 4(1):25–43, 2002.

[21] Pao-Long Chang and Chin-Tsai Lin. On the effect of centralization on expected costs in a multi-location newsboy problem. *Journal of the Operational Research Society*, 42(11):1025–1030, 1991.

[22] Daniel Chemla, Frédéric Meunier, and Roberto Wolfler Calvo. Bike sharing systems: Solving the static rebalancing problem. *Discrete Optimization*, 10(2):120–146, 2013.

[23] Huan Neng Chiu and Hau Lieng Huang. A multi-echelon integrated jit inventory model using the time buffer and emergency borrowing policies to deal with random delivery lead times. *International journal of production research*, 41(13):2911–2931, 2003.

[24] Albert Corominas, Jordi Olivella, and Rafael Pastor. A model for the assignment of a set of tasks when work performance depends on experience of all tasks involved. *International Journal of Production Economics*, 126(2):335–340, 2010.

[25] Teodor G Crainic, Francesca Maggioni, Guido Perboli, and Walter Rei. Reduced cost-based variable fixing in two-stage stochastic programming. *Annals of Operations Research*, pages 1–37, 2017.

[26] Ezey M Dar-El. *Human learning: From learning curves to learning organizations*, volume 29. Springer Science & Business Media, 2013.

[27] Sharon Datner, Tal Raviv, Michal Tzur, and Daniel Chemla. Setting inventory levels in a bike sharing network. *Transportation Science*, 2017.

[28] Mauro Dell'Amico, Eleni Hadjicostantinou, Manuel Iori, and Stefano Novellani. The bike sharing rebalancing problem: Mathematical formulations and benchmark instances. *Omega*, 45:7–19, 2014.

[29] Mauro Dell'Amico, Manuel Iori, Stefano Novellani, Thomas Stützle, et al. A destroy and repair algorithm for the bike sharing rebalancing problem. *Computers & Operations Research*, 71:149–162, 2016.

[30] Gary D Eppen. Noteeffects of centralization on expected costs in a multi-location newsboy problem. *Management science*, 25(5):498–501, 1979.

[31] Güneş Erdoğan, Gilbert Laporte, and Roberto Wolfler Calvo. The static bicycle relocation problem with demand intervals. *European Journal of Operational Research*, 238(2):451–457, 2014.

[32] Alan L Erera, Martin Savelsbergh, and Emrah Uyar. Fixed routes with backup vehicles for stochastic vehicle routing problems with time constraints. *Networks*, 54(4):270–283, 2009.

[33] Wei Fan. Optimizing strategic allocation of vehicles for one-way car-sharing systems under demand uncertainty. *Journal of the Transportation Research Forum*, 53:7–20, 2014.

[34] Xuehao Feng, Ilkyeong Moon, and Kwangyeol Ryu. Warehouse capacity sharing via transshipment for an integrated two-echelon supply chain. *Transportation Research Part E: Logistics and Transportation Review*, 104:17–35, 2017.

[35] Iris A Forma, Tal Raviv, and Michal Tzur. A 3-step math heuristic for the static repositioning problem in bike-sharing systems. *Transportation research part B: methodological*, 71:230–247, 2015.

[36] Andrew Gelman. Scaling regression inputs by dividing by two standard deviations. *Statistics in medicine*, 27(15):2865–2873, 2008.

[37] Supriyo Ghosh, Pradeep Varakantham, Yossiri Adulyasak, and Patrick Jaillet. Dynamic repositioning to reduce lost demand in bike sharing systems. *Journal of Artificial Intelligence Research*, 58:387–430, 2017.

[38] Gurobi. Gurobi optimization. URL `www.gurobi.com`.

[39] Christian Heimerl and Rainer Kolisch. Work assignment to and qualification of multi-skilled human resources under knowledge depreciation and company skill level targets. *International Journal of Production Research*, 48(13):3759–3781, 2010.

[40] Yale T Herer and Ayelet Rashit. Lateral stock transshipments in a two-location inventory system with fixed and joint replenishment costs. *Naval Research Logistics (NRL)*, 46(5):525–547, 1999.

[41] Yale T Herer and Michal Tzur. The dynamic transshipment problem. *Naval Research Logistics (NRL)*, 48(5):386–408, 2001.

[42] Yale T Herer and Michal Tzur. Optimal and heuristic algorithms for the multi-location dynamic transshipment problem with fixed transshipment costs. *IIE Transactions*, 35(5):419–432, 2003.

[43] Yale T Herer, Michal Tzur, and Enver Yücesan. The multilocation transshipment problem. *IIE transactions*, 38(3):185–200, 2006.

[44] Mike Hewitt, Austin Chacosky, Scott E Grasman, and Barrett W Thomas. Integer programming techniques for solving non-linear workforce planning models with learning. *European Journal of Operational Research*, 242(3):942–950, 2015.

[45] Sin C Ho and WY Szeto. Solving a static repositioning problem in bike-sharing systems using iterated tabu search. *Transportation Research Part E: Logistics and Transportation Review*, 69: 180–198, 2014.

[46] Stephen B Hulley, Steven R Cummings, Warren S Browner, Deborah G Grady, and Thomas B Newman. *Designing clinical research*. Lippincott Williams & Wilkins, 2013.

[47] International Trade Administration, 2017.

[48] Mohamad Y Jaber. Learning and forgetting models and their applications. *Handbook of industrial and systems engineering*, pages 30–1, 2006.

[49] Huan Jin, Mike Hewitt, and Barrett W Thomas. Workforce grouping and assignment with learning-by-doing and knowledge transfer. *International Journal of Production Research*, pages 1–15, 2018.

[50] Peter Kall and Stein W Wallace. *Stochastic Programming*. Springer, 1994.

[51] Michal Kaut and Stein W Wallace. Evaluation of scenario-generation methods for stochastic programming. *Pacific Journal of Optimization*, 3(2):257–271, 2007.

[52] Alan J King and Stein W Wallace. *Modeling with stochastic programming*. Springer Science & Business Media, 2012.

[53] Andreas Klose. Single-sink fixed-charge transportation: Applications and exact solution algorithms. *Working Papers, Department of Mathematical Sciences, University of Aarhus*, 2006(5), 2006.

[54] Chung-Cheng Lu. Robust multi-period fleet allocation models for bike-sharing systems. *Networks and Spatial Economics*, 16(1):61–82, 2016.

[55] Francesca Maggioni and Georg Ch Pflug. Bounds and approximations for multistage stochastic programs. *SIAM Journal on Optimization*, 26(1):831–855, 2016.

[56] Francesca Maggioni and Stein W Wallace. Analyzing the quality of the expected value solution in stochastic programming. *Annals of Operations Research*, 200(1):37–54, 2012.

[57] Francesca Maggioni, Michal Kaut, and Luca Bertazzi. Stochastic optimization models for a single-sink transportation problem. *Computational Management Science*, 6(2):251–267, 2009.

[58] Francesca Maggioni, Elisabetta Allevi, and Marida Bertocchi. Monotonic bounds in multistage mixed-integer stochastic programming. *Computational Management Science*, 13(3):423–457, Jul 2016.

[59] Francesca Maggioni, Florian A Potra, and Marida Bertocchi. A scenario-based framework for supply planning under uncertainty: stochastic programming versus robust optimization approaches. *Computational Management Science*, 14(1):5–44, 2017.

[60] Kenneth A Marentette, Alan W Johnson, and Lisa Mills. A measure of cross-training benefit versus job skill specialization. *Computers & Industrial Engineering*, 57(3):937–940, 2009.

[61] Joseph B Mazzola and Kevin F McCardle. A bayesian approach to managing learning-curve uncertainty. *Management Science*, 42(5):680–692, 1996.

[62] Joseph B Mazzola and Kevin F McCardle. The stochastic learning curve: Optimal production in the presence of learning-curve uncertainty. *Operations Research*, 45(3):440–450, 1997.

[63] Joseph B Mazzola, Alan W Neebe, and Christopher M Rump. Multiproduct production planning in the presence of work-force learning. *European Journal of Operational Research*, 106(2):336–356, 1998.

[64] John K McCreery and Lee J Krajewski. Improving performance using workforce flexibility in an assembly environment with learning and forgetting effects. *International Journal of Production Research*, 37(9):2031–2058, 1999.

[65] John K McCreery, Lee J Krajewski, G Keong Leong, and Peter T Ward. Performance implications of assembly work teams. *Journal of Operations Management*, 22(4):387–412, 2004.

[66] Stefan Minner and Edward A Silver. Evaluation of two simple extreme transshipment strategies. *International Journal of Production Economics*, 93:1–11, 2005.

[67] Stefan Minner, Edward A Silver, and David J Robb. An improved heuristic for deciding on emergency transshipments. *European Journal of Operational Research*, 148(2):384–400, 2003.

[68] Seyed MJ Mirzapour Al-e hashem, Yacine Rekik, and Ebrahim Mohammadi Hoseinhajlou. A hybrid l-shaped method to solve a bi-objective stochastic transshipment-enabled inventory routing problem. *International Journal of Production Economics*, 2017.

[69] David A Nembhard and Frank Bentefouet. Parallel system scheduling with general worker learning and forgetting. *International Journal of Production Economics*, 139(2):533–542, 2012.

[70] Jordi Olivella, Albert Corominas, and Rafael Pastor. Task assignment considering cross-training goals and due dates. *International Journal of Production Research*, 51(3):952–962, 2013.

[71] Fredrik Olsson. An inventory model with unidirectional lateral transshipments. *European Journal of Operational Research*, 200(3):725–732, 2010.

[72] Deniz Özdemir, Enver Yücesan, and Yale T Herer. Multi-location transshipment problem with capacitated transportation. *European Journal of Operational Research*, 175(1):602–621, 2006.

[73] Colin Paterson, Gudrun Kiesmüller, Ruud Teunter, and Kevin Glazebrook. Inventory models with lateral transshipments: A review. *European Journal of Operational Research*, 210(2):125–136, 2011.

[74] Python Software Foundation. Python. URL `www.python.org`.

[75] Ruwen Qin, David A Nembhard, and Walter L Barnes II. Workforce flexibility in operations management. *Surveys in Operations Research and Management Science*, 20(1):19–33, 2015.

[76] Tal Raviv and Ofer Kolka. Optimal inventory management of a bike-sharing station. *IIE Transactions*, 45(10):1077–1093, 2013.

[77] Tal Raviv, Michal Tzur, and Iris A Forma. Static repositioning in a bike-sharing system: models and solution approaches. *EURO Journal on Transportation and Logistics*, 2(3):187–229, 2013.

[78] Robert Regue and Will Recker. Proactive vehicle routing with inferred demand to solve the bikesharing rebalancing problem. *Transportation Research Part E: Logistics and Transportation Review*, 72:192–209, 2014.

[79] Roberto Roberti, Enrico Bartolini, and Aristide Mingozzi. The fixed charge transportation problem: An exact algorithm based on a new integer programming formulation. *Management Science*, 61(6):1275–1291, 2014.

[80] Lawrence W Robinson. Optimal and approximate policies in multiperiod, multilocation inventory models with transshipments. *Operations research*, 38(2):278–295, 1990.

[81] Beate Rottkemper, Kathrin Fischer, and Alexander Blecken. A transshipment model for distribution and inventory relocation under uncertainty in humanitarian operations. *Socio-Economic Planning Sciences*, 46(1):98–109, 2012.

[82] Kevin Ryan, Shabbir Ahmed, Santanu S Dey, and Deepak Rajan. Optimization driven scenario grouping. *Available at Optimization-Online http://www. optimization-online. org/DB_FILE/2016/03/5366. pdf*, 2016.

[83] Serpil Sayın and Selçuk Karabatı. Assigning cross-trained workers to departments: A two-stage optimization model to maximize utility and skill improvement. *European Journal of Operational Research*, 176(3):1643–1658, 2007.

[84] Qingning Shen, Feng Chu, and Haoxun Chen. A lagrangian relaxation approach for a multi-mode inventory routing problem with transshipment in crude oil transportation. *Computers & Chemical Engineering*, 35(10):2113–2123, 2011.

[85] Jannes Slomp and Eric Molleman. Cross-training policies and team performance. *International Journal of Production Research*, 40(5):1193–1219, 2002.

[86] Statsmodels-developers. Statsmodels statistics in python. URL `www.statsmodels.org`.

[87] James H Stock and Mark W Watson. *Introduction to Econometrics*. Pearson, 2007.

[88] Silviya Valeva, Mike Hewitt, Barrett W Thomas, and Kenneth G Brown. Balancing flexibility and inventory in workforce planning with learning. *International Journal of Production Economics*, 2016.

[89] ACC van Wijk, IJBF Adan, and GJ van Houtum. Optimal lateral transshipment policies for a two location inventory problem with multiple demand classes. *European Journal of Operational Research*, 2018.

[90] Stein W Wallace and William T Ziemba. *Applications of Stochastic Programming*. SIAM, 2005.

[91] Kwan Eng Wee and Maqbool Dada. Optimal policies for transshipping inventory in a retail network. *Management science*, 51(10):1519–1533, 2005.

[92] World Economic Forum. https://www.weforum.org/agenda/2016/02/the-global-bike-sharing-boom-why-cities-love-cycling-schemes, 2016.

[93] Enver Yücesan et al. Stochastic optimization for transshipment problems with positive replenishment lead times. *International Journal of Production Economics*, 135(1):61–72, 2012.

List of figures

2.1 Total cost comparison between the optimal (or near optimal) solution and the Rolling horizon approach (RH) for every instance of Case (1) with $W = 1$ and deadline $T = 3$ and unit inventory costs equal to 5% and 20% of the unit item price. 36
2.2 Total cost comparison between the optimal (or near optimal) solution and the Rolling horizon approach (RH) for every instance of Case (1) with $W = 2$ and deadline $T = 3$ and unit inventory costs equal to 5% and 20% of the unit item price. 37

3.1 Illustration of the two-stage decision-making process 47
3.2 Congestion and starvation average frequencies in the real case for year 2016, considering the months from May to August and the time interval between 6 a.m. and 11.59 a.m. . 54
3.3 San Francisco bike sharing system 55
3.4 Empirical distributions and Monte Carlo sampling for two different stations 56
3.5 In sample and out-of-sample analysis 58
3.6 Distributions of bikes to stations in the EV and SP solutions 61
3.7 Number of allocated bikes vs demand variance. 62

4.1 Illustration of production setting 75
4.2 Illustration of the sequence of decisions and events 78
4.3 Learning rate distributions 84
4.4 Different factor levels combinations 84
4.5 Stochastic solution analysis for an instance 90
4.6 Comparing scheduling and production decisions 91
4.7 Coefficients for *na* regression. 92
4.8 Coefficients for *np* regression. 93
4.9 Coefficients for CTI, PI regressions with p-value $< .05$ 94

5.1 A refinement chain with disjoint groups 101
5.2 An alternative refinement chain with disjoint groups (1) 101

5.3 An alternative refinement chain with disjoint groups (2) 102
5.4 An alternative refinement chain with disjoint groups (3) 102
5.5 Scenario objective function values for the Bikesharing Allocation and Rebalancing Problem for an increasing number of scenarios. 106
5.6 Scenario objective function values for the $SSLP\_10\_50\_50$ 111

List of tables

1.1 Transshipment and backordering fixed and unit costs 11

1.2 Average values for the stochastic solution analysis indicators for every special case with two retailers 13

1.3 Values for the stochastic solution analysis indicators for every special case with four retailers and "Medium" cost level 13

2.1 Shipment, transshipment and backordering fixed and variable costs for every case and instance. 28

2.2 Case (1): Summary statistics 29

2.3 Case (2): Summary statistics 29

2.4 Case (3): Summary statistics for subcases a), b), c) 30

2.5 Case (1): Optimal value of the first-stage variables x^{0*} and of the total cost, for different time horizons, different unit inventory costs and different vehicle capacities. 31

2.6 Optimal value of the first-stage variables x^{0*} and total cost and CPU time (in seconds) for the Rolling horizon approach with $W = 1$ and $W = 2$, respectively, for instances belonging to Case (1) with deadline $T = 3$ (i.e. four-stage) and unit inventory costs equal to 5% of the item price. 34

2.7 Comparison in terms of objective value and CPU seconds of the solution provided by the Rolling horizon approach, with $W = 1$ and $W = 2$, respectively, with respect to the optimal (or near-optimal) solution, for instances belonging to Case (1) with deadline $T = 3$ (i.e. four-stage) and unit inventory costs equal to 5% of the item price. 34

2.8 Optimal value of the first-stage variables x^{0*} and total cost and CPU time (in seconds) for the Rolling horizon approach with $W = 1$ and $W = 2$, respectively, for instances belonging to Case (1) with deadline $T = 3$ (i.e. four-stage) and unit inventory costs equal to 20% of the item price. 35

2.9 Comparison in terms of objective value and CPU seconds of the solution provided by the Rolling horizon approach, with $W = 1$ and $W = 2$, respectively, with respect to the optimal (or near-optimal) solution, for instances belonging to Case (1) with deadline $T = 3$ (i.e. four-stage) and unit inventory costs equal to 20% of the item price. 35

2.10 Optimal value of the first-stage variables x^{0*} and total cost and CPU time (in seconds) for the Rolling horizon approach with $W = 1$, for instances belonging to case (1) with deadline $T = 4$ (i.e. five-stage model) and unit inventory costs equal to 5% of the item price. 38

2.11 Comparison in terms of objective value and CPU seconds of the solution provided by the Rolling horizon approach, with $W = 1$, with respect to the optimal (or near-optimal) solution, for instances belonging to Case (1) with deadline $T = 4$ (i.e. five-stage) and unit inventory costs equal to 5% of the item price. 38

3.1 Simulation-based comparison of solutions to stochastic and deterministic problems. . . . 60

3.2 Objective function and computational time results comparison of the stochastic program and the SBH's. 63

3.3 Simulation results comparison of the stochastic program and the SBH's. 64

3.4 Comparison of implemented plan and plan from stochastic program 65

3.5 Comparison of implemented plan and plan from stochastic program, when allocating same number of bikes. 66

4.1 Parameter values 84

4.2 Instance factors 85

4.3 In-sample analysis results 85

4.4 Out-of-sample analysis results 86

4.5 Summary regression results 91

5.1 Comparison of lower bounds found through model (5.8)-(5.17), Maggioni and Pflug [55] (2016) and Ryan, Ahmed, Dey, and Rajan [82] (2016) for the Bikesharing problem with 16 scenarios. 108

5.2 Comparison of lower bounds found through model (5.8)-(5.17), Maggioni and Pflug [55] (2016) and Ryan, Ahmed, Dey, and Rajan [82] (2016) for the Bikesharing problem with 100 scenarios. 109

5.3 Comparison of lower bounds found through model (5.8)-(5.17), Maggioni and Pflug [55] (2016) and Ryan, Ahmed, Dey, and Rajan [82] (2016) for the Bikesharing problem with 140 scenarios. 110
5.4 Comparison of lower bounds for the $SSLP\_10\_50\_50$, found through model (5.8)-(5.17), Maggioni and Pflug [55] (2016) and Ryan, Ahmed, Dey, and Rajan [82] (2016). 112
2.b.1 Congestion and starvation relative frequencies by method of determining initial bike requirement 119
3.b.1 Detailed linear regression results for na 124
3.b.2 Detailed linear regression results for np 125
3.b.3 Detailed linear regression results for CTI 126
3.b.4 Detailed linear regression results for PI 127