University Paris-Saclay,
M2 – Optimization, Academic year 2016-2017
LRI – Laboratoire de Recherche en Informatique

REPORT OF INTERNSHIP

"Stochastic lot-sizing for remanufacturing: a multi-stage stochastic integer programming approach"

Author:
Franco Quezada Valenzuela

Supervisors:
Dr. Céline Gicquel
Dr. Safia Kedad-Sidhoum

17 August, 2017

Declaration

I, Franco Quezada Valenzuela, declare that this report of internship titled,"Stochastic lot-sizing for remanufacturing system: a multi-stage stochastic integer programming approach" and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for Master 2 degree - Optimisation program of University Paris Saclay, as the internship project done at LRI-Laboratoire de Recherche en Informatique, funded as a part of a one-year project supported by the Gaspard Monge program.
- Where any part of this report has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this report is entirely my own work.
- I have acknowledged all main sources of help.

Signed: _____

Date: _____

Acknowledgement

Thank all those who contributed in some way to the realization of this research. Specially, I would like to express my profound gratitude to my supervisors Dr. Cline Gicquel and Dr. Safia Kedad-Sidhoum for their time, advice and support. I also would like to give my thanks to the researchers and other intern-students in team ROCS, as well as all people and staffs at LRI-Laboratoire de Recherche en Informatique for their helps throughout my internship. Finally, I would like to thank my parents and friends, who always support me through in this arduous way.

This research was partially supported by the supercomputing infrastructure of the NLHPC (ECM-02) and was partially funded by the BECAS CHILE program, CONICYT and the Gaspard Monge program (PGMO), FMJH.

For each one of them, Thank you!

Abstract

Re-manufacturing systems are defined as sets of processes transforming end-of-life products (returned products) into like-new products, once again usable by customers, mainly by rehabilitating damaged components. Within this project, we consider a re-manufacturing system involving three production echelons: (1) *disassembling* the used products brought back by customers, (2) *refurbishing* the recovered parts and (3) *reassembling* them into like-new finished products. We seek to plan the production activities on this system over a multi-period horizon. This leads to the formulation of a multi-echelon lot-sizing problem with lost sales, mathematically expressed as a Mixed-Integer Linear Programming model. We consider the stochastic variant of the problem, in which the uncertainties on the quantity and quality of returned products and the customers' demand are explicitly taken into account. We propose a multi-stage stochastic programming approach relying on scenario trees to represent the uncertain information structure. We develop a Cut-and-Branch algorithm in order to solve the resulting mixed-integer linear program to optimality. This algorithm relies on a new set of tree valid inequalities obtained by combining valid inequalities previously known for each individual scenario of the scenario tree into inequalities valid for a subset of scenarios, i.e. for subtrees of the scenario tree. These inequalities are added to the problem formulation using a cutting-plane generation procedure based either on an exact or on a heuristic resolution of the corresponding separation problem. Some computational results carried out on medium-size randomly generated instances of the problem are provided. They show that the proposed Cut-and-Branch algorithm is capable of significantly improving the quality of the solutions obtained within a limited computation time as compared to the use of a stand-alone mathematical solver.

Keywords: *Stochastic lot-sizing, re-manufacturing system, valid inequalities, polyhedral approach, Cut-and-Branch algorithm, multi-stage stochastic programming, scenario tree*

Contents

Chapter 1

Introduction

Nowadays, industrial companies face an increasing pressure from customers and governments to become more environmentally responsible and to mitigate the environmental impact of their products. One way of achieving this objective is to re-manufacture the products once they have reached their end-of-life. Remanufacturing can be defined as a set of processes transforming end-of-life products (used products/ returns) into like-new finished products, once again usable by customers, mainly by rehabilitating damaged components. By reusing the materials and components embedded in used products, it both contributes in reducing pollution emissions and natural resource consumption.

Remanufacturing systems can be found in a variety of industrial sectors: car production, electrical goods, aircraft engines, computers, smart-phones, chemical products, etc. (see Lund (1984)). They are usually complex systems encompassing a variety of activities such as used products collecting, testing and sorting, disassembling, refurbishing, reassembling and redistributing, etc.

In this work, we focus on three of these keys processes: disassembly, refurbishing and reassembly and aim at optimizing the production planning for the corresponding three-echelon system over a multi-period horizon. Production planning involves making decisions about the production level (i.e. which products and how much of them should be made), the timing (i.e. when the products should be be made) and the resources to be used. Production planning in remanufacturing includes making decisions on used products returned by customers, such as, how much and when used products must be disassembled, refurbished or reassembled in order to build new or like new products. The main objective in production planning is to meet customers demand for the company products in the most cost-effective way.

Lot-sizing problems arise in production situations which involve set-up operations (tool changes, machine calibration, machine installation, etc.) incurring fixed set-up costs. As a naive perception, to reduce these set-up costs, production should be run using large lot sizes. However, this generates de-synchronized patterns between the customers demand and the production plan leading to costly high levels of inventory. Lot-sizing models thus aim at reaching the best possible trade-off between minimizing the set-up costs and minimizing the inventory holding costs, under constraints on customer demand satisfaction and technical limitations of the system. They are usually modeled as mixed-integer linear programs.

As compared to 'classical' manufacturing systems producing end products from virgin raw materials and new components, remanufacturing systems involve several complicating characteristics, among which is a high level of uncertainty in the input data needed to make planning decisions (Guide (2000)). This is mainly due to a lack of control on return flows of used products, both in term of quantity and quality, and to the difficulty to forecast the demand for new (or like-new) products. Even in cases where companies apply special policies to collect the used products from customers,

i.e, product life-cycle contracts or collecting incentives, etc., these parameters still remain difficult to accurately predict. To deal with this uncertainty, we aim at developing a multi-stage stochastic programming approach in which the uncertain input parameters are modeled as discrete random variables whose value is unfolded little by little as time progresses.

1.1 Problem description

We consider a remanufacturing system involving three production echelons: Disassembling, Refurbishing, and Reassembling, where used products can be discarded before being disassembled and a proportion of the parts recovered from the disassembly process cannot be renewed due to their low quality state. We take our focus on optimizing the production planning for such a system over a multi-period horizon under uncertain input data which implies solving a stochastic lot-sizing problem. More specifically, the problem is to determine the lot sizes for each production process (i.e. the disassembly, refurbishing and reassembly processes) in each period in order to satisfy the dynamic demand of re-manufactured products over the planning horizon. In other words, the main decisions include how much and when to disassemble used products, how much and when to discard used products, how much and when to refurbish items and how much and when to reassemble items into re-manufactured products.

We thus consider a multi-item remanufacturing system (illustrated in Figure 1.1) where the main production echelons are defined as follows:

- Echelon 1: disassembly of used products returned by customers into a series of recoverable parts (items).
- Echelon 2: refurbishing of recoverable parts providing well-functioning serviceable parts.
- Echelon 3: reassembly of serviceable parts into re-manufactured products considered as-good-as-new which are then sold to customers.

We use the following modeling assumptions:

- There is a single type of returned and re-manufactured product.
- There is a limited quantity of used products returned in each period.
- Used products and recoverable parts can be discarded in order to keep low inventory.
- The yields of the refurbishing and reassembly processes are deterministic and time invariant.
- The yield of the disassembly process, i.e the number of recoverable parts, is part dependent and time-dependent. This reflects variations in the quality of returned products which impact the proportion of parts that can be recovered by disassembly.
- In case the amount/quality of returned products does not enable the system to satisfy the customer demand on time, the corresponding demand is lost. The unsatisfied demand incurs a high penalty cost to account for the loss of customers goodwill.

Notice that we assume in the echelon 2, there is a dedicated production process for the refurbishing of each type of recoverable part. Furthermore, our model incorporates a complicating feature. We namely allow some used products to be discarded before being disassembled: this option might be useful in case more used products are returned than what is requested to satisfy

the demand for remanufactured products. Similarly, we allow to discard some of the recoverable parts obtained from the disassembly process. In case there is a strong unbalance between the part-dependent disassembly yields, this option might be used in a production plan to avoid an unnecessary accumulation in inventory of the 'easy-to-recover' parts.

We seek to build a production plan for such a system over a multi-period horizon minimizing the total remanufacturing costs. It comprises the set-up costs for all production processes, the inventory holding costs for all products involved in the system, the costs for discarding used products and recoverable parts and the lost sales costs penalizing the non-satisfaction of the customer demand.

Note that a similar system was studied by Vu et al. (2017). However, Vu *et al.* assumed that all parts obtained by disassembling could be recovered and discarding used products and recoverable parts were forbidden. Thus, in the present work, we extend the work of Vu et al. (2017) to a more complex remanufacturing system in order to better model real-life situations and to improve the practical relevance of the obtained productions plans.

Similarly to the work of Vu et al. (2017), we consider uncertainties on the quantity of returned products, on the demand for remanufactured products and the production costs. Moreover, we extend their work by incorporating in the model uncertainties on the quality of the returned products, which is represented by stochastic disassembly yields. To investigate the resulting stochastic problem, we discuss a multi-stage stochastic programming approach in which uncertainty is represented by a scenario tree. This gives rise to the formulation of large-size mixed integer linear program which we seek to solve to optimality through a Cut-and-Branch algorithm.

1.2 Related literature

In order to highlight the main contributions of this work, we present an overview of the related previous works. We consider three main research areas: production planning in remanufacturing systems, lot-sizing for remanufacturing and stochastic lot-sizing.

1.2.1 Production planning in remanufacturing

Lund (1984) defines remanufacturing as "an industrial process in which worn out products are restored to like new condition". Fleischmann et al. (1997) provided one of the first literature review on quantitative models for reverse logistics. Guide et al. (1999) and Guide (2000) extensively discussed the fact that production planning and control activities are more complex for remanufacturing firms due to uncertainties from stochastic product returns, imbalances in return and demand rates, and the unknown condition of used products. Ilgin and Gupta (2010) provided an overview of the related literature to Enviromentally Conscious Manufacturing and Product Recovery (ECMPRO) and discussed the new research areas developed in closed-loop supply chains, remanufacturing and disassembly. Furthermore, they analyzed the gap in the research field between this new overview and the general framework in ECMPRO provided by Gungor and Gupta (1999). Junior and Filho (2012) provided a new literature review about production planning and control for remanufacturing systems and analyzed whether the gap identified by Guide (2000) was fully investigated. Finally, Lage Junior and Godinho Filho (2016) explored similarities and differences between theoretically listed needs and real cases and concluded that difficulties related to uncertainties have not been addressed to date.

1.2.2 Lot-sizing in remanufacturing systems

We will review the papers based on two classification criteria: the structure of the remanufacturing system and the presence/absence of uncertainty in the production planning model.

The use of lot-sizing in remanufacturing system is relatively new in the literature. Research done in this field can first be categorized according to three main types of remanufacturing systems:

- Single-echelon remanufacturing/manufacturing system: the different production steps (or echelons) required to transform a returned product into a remanufactured product are represented in a simplified aggregated manner via a single production echelon but the remanufacturing system is often coupled with a manufacturing system. This hybrid use of remanufacturing and manufacturing systems can be composed in different variants: either customers' demand on re-manufactured products and newly manufactured products can be mutually substituted (see Teunter et al. (2006) and Macedo et al. (2016)), or there is one-way substitution assumption (see Pineyro and Viera (2010)), i.e. the demand for remanufactured products can be satisfied by new products but the opposite is not allowed. Some solution approaches can be found in Sahling (2013), Retel Helmrich et al. (2014) and Naeem et al. (2013).

- Multi-echelon disassembly system: the different production steps (or echelons) required to disassemble a returned product into as-good-as new parts are explicitly modeled but the reassembly of parts into remanufactured products is not considered. Disassembling processes can be considered as an alternative components supplying source (see Kim et al. (2006)) or as a stand alone operation (see Sung and Jeong (2014) and (Junior and Godinho Filho, 2017)).

- Multi-echelon disassembly-reassembly system: the different production steps (or echelons) required to transform a returned product into a remanufactured product are explicitly modeled. An example of using this framework is Jayaraman (2006). Kang et al. (2011) considered multi-product type problem in which disassembly levels may be different even for products of the same type. Hashemi et al. (2014) studied a capacitated integrated system manufacturing/remanufacturing in which returned products are disassembled into components and used componentes can be either disposed of as required or disassembled into parts. On the other hand, Wang and Huang (2013) studied a demand-driven product disassembly, recycling, and remanufacturing system with multi-period and multi-product and a two-stage robust programming approach to demand-driven disassembly planning is proposed. Note that *our lot-sizing for remanufacturing system can be categorized into this type.*

Regarding to the presence or absence of uncertainty in the problem modeling, we note that most previously published works deal with deterministic optimization problems (Teunter et al. (2006), Pineyro and Viera (2010),Kang et al. (2011), Sahling (2013), Kim et al. (2006), Jayaraman (2006), Hashemi et al. (2014), Retel Helmrich et al. (2014), Sung and Jeong (2014)). However, a few recent papers explicitly take into account uncertainties in their problem modeling. Naeem et al. (2013) and Wang and Huang (2013) consider stochastic demand and/or return quantity and Macedo et al. (2016) is the first work taking into account stochastic demand, return and set-up cost simultaneously for hybrid manufacturing/remanufacturing systems. On the other hand, Junior and Godinho Filho (2017) consider stochastic routing for a master disassembly scheduling model.

In terms of solution approaches, it seems that two main types of solution approaches have been investigated up to now:

- Stochastic dynamic programming: Naeem et al. (2013) and Junior and Godinho Filho (2017).

- Two-stage stochastic programming : Macedo et al. (2016) and Wang and Huang (2013).

We consider a multi-echelon system with both disassembly and reassembly operations and explicitly consider in our problem modeling the uncertainty on the demand, the return quantity and quality and the production costs. Our work is therefore closely related to the one of Wang and Huang (2013). However, Wang and Huang develop a two-stage stochastic programming approach in which all production decisions are first-stage decisions, i.e. are made once and for all before the realization of the uncertain parameters, and cannot be updated later on. In contrast, we seek to develop a multi-stage stochastic programming approach in which production decisions can be made little by little as new information about the uncertain parameters becomes available. To the best of our knowledge, this is the first attempt at proposing a multi-stage stochastic programming approach for a stochastic multi-echelon lot-sizing problem arising in a remanufacturing context.

1.2.3 Stochastic Programming for Lot-sizing

Investigating lot-sizing problems under stochastic input data is an active research area in production planning field. We can classify the relating papers according to their modeling of the decision process.

Single-stage models seek to build a production plan for the whole horizon beforehand, and do not consider the possibility to adjust it, once new information becomes available. The most commonly found option is to consider that demand is stochastic and non-stationary. Continuous probability distribution of the demand is found in Vargas (2009), Piperagkas et al. (2012) and Tempelmeier and Herpers (2011). On the other hand, Küçükyavuz (2012) studies chance-constrained programs with discrete probability distributions for the demand.

In contrast, multi-stage models allow us to postpone some of decisions to a later point in time, when new information on uncertain parameters becomes available. This type of model relies on the fact that only part of the decisions have to be made prior to the realization of the demand and that we may have the possibility to revise the production plan several times during the planning horizon in response to the outcomes of the random demand. A first line of research corresponds to the case where the decisions to be made prior the realization of the demand (first-stage decisions) pertain to the periods where production will occur (Tempelmeier (2007), Tarim and Kingsman (2004), Bookbinder and Tan (1988)). A second line of research considers that the only first-stage decisions to be made are the ones related to the first period of the planning horizon and that all decisions related to the following periods can be decided upon after the demand has realized. This type of model usually relies on a description of the uncertainty through a scenario tree and seeks to define the best possible production policy (Di Summa and Wolsey (2008), Guan et al. (2006a), Haugen et al. (2001) and Zhang et al. (2014).

We now focus on multi-stage stochastic programming models based on scenario trees as our work belongs to this line of research and provide an overview of the different types of solution approaches which have been proposed to solve the resulting large-size MILP.

A first class of solution approaches are exact solution approaches relying either on polynomial-time algorithms based on dynamic programming (see Guan and Miller (2008), Huang and KüçüKyavuz (2008) and Zhao and Guan (2014)) or on linear programming relaxation strengthening, which can use valid inequalities (see Guan et al. (2006b), Guan et al. (2009), Di Summa and Wolsey (2008) and Zhang et al. (2014)) and/or extended formulations (see Zhao and Guan (2014)). On the other hand, the second class corresponds to heuristic/metaheuristic solution approaches such as, 'progressive hedging' combined with 'tabu search' (see Haugen et al. (2001)), 'fix and relax' method (see Beraldi et al. (2006) and Brandimarte (2006)), Lagrangian relaxation approach (see Tang et al. (2012)) and updating scenario heuristic (see Zanjani et al. (2013)), amongst others.

In the present work, we investigate a multi-stage stochastic programming approach based on a

scenario tree for a stochastic lot-sizing problem. Moreover, we seek to develop an exact solution approach based on the strengthening of the linear relaxation of the problem through the use of a set of valid inequalities. Our work is thus closely related to the ones of Guan et al. (2006b);Guan et al. (2009); Di Summa and Wolsey (2008) and Zhang et al. (2014). However, all these papers focus on single-echelon production systems and do not consider used product returns nor lost sales. In contrast, we study a multi-echelon production system involving product returns and lost sales. More precisely, we seek to extend the work of Guan et al. (2009) to this more complex but also more realistic production planning setting.

1.3 Contributions

Most of previous works dealt with simple lot-sizing in remanufacturing production planning problem, i.e. with single item, single echelon process or single period, and consider a few of the complicating features in remanufacturing systems enumerated by Guide et al. (1999); Guide (2000). Vu et al. (2017) studied a more complex problem considering multi-item multi-echelon stochastic lot-sizing in remanufacturing system problem with lost sales over a multi-period horizon. We propose an extension of this problem taking into account two additional complicating features, i.e. (1) used products can be discarded before the disassembly process and a proportion of recoverable parts from disassembly process can be discarded before the refurbishing process in order to avoid useless inventory build-up, (2) a stochastic proportion of the product items recovered from disassembly cannot be refurbished to create new products due to their low quality.

We first provide a mathematical formulation for the deterministic variant of the problem. More specifically, a mixed integer linear programming formulation based on an 'intuitive definition' of the inventory variables for each period is provided for the deterministic case. Then, we investigate the stochastic version of the problem and propose a multi-stage stochastic programming approach in which the uncertainty is represented via a scenario tree. This leads us to the formulation of a large-size mixed integer programming model.

Thanks to the use of the echelon inventory concept, the problem can be reformulated into a series of stochastic single-echelon lot-sizing subproblems linked by inter-echelon coupling constraints. We then seek to strengthen the linear relaxation of each of these stochastic subproblems by extending valid inequalities known for their deterministic counterpart to the stochastic case. This is achieved by extending the work of Guan et al. (2009) who proposed to generate cutting planes for multi-stage stochastic integer programs by combining "Path inequalities" that are valid for the deterministic case and for individual scenarios of the scenario tree in order to obtain stronger "Tree inequalities" involving multiple scenarios of the scenario tree. Note that Guan et al. (2009) applied this idea to solve a single-item single-echelon stochastic lot-sizing without product returns nor lost-sales whereas we seek to exploit it to solve a more complex and realistic lot-sizing problem involving multiple production echelons, product returns and lost-sales. Moreover, we would like to point out that Vu et al. (2017) only used path inequalities in their Cut& Branch algorithm whereas our cutting-plane generation algorithm exploits both path and tree inequalities, allowing us to improve the formulation strengthening. This can be understood as one of the major contributions of this master thesis.

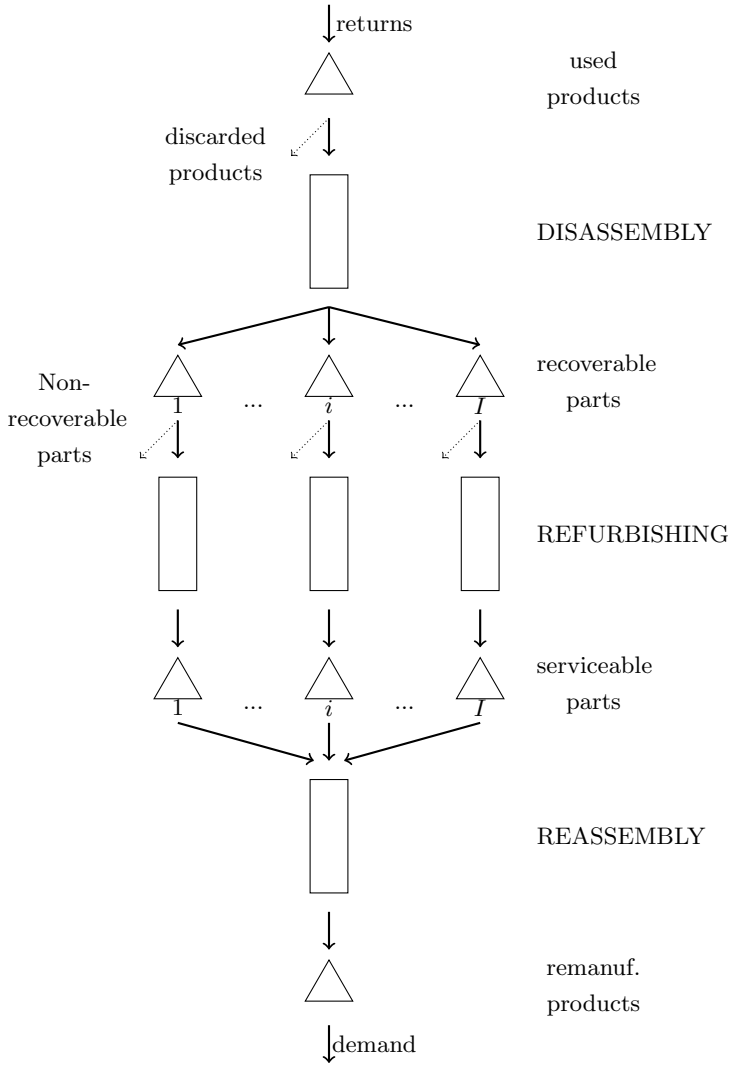

Figure 1.1: Studied remanufacturing system

1.4 Outline of the report

The remainder of the manuscript is organized as follows. Chapter 2 is divided into two subsections. Subsection 2.1 is devoted to the mathematical formulation for the problem in its deterministic variant. We thus assumed that all input parameters are known at the time the production plans is built. Subsection 2.2 is dedicated to the stochastic variant of the problem. We assume that some of the input parameters follow a stochastic process whose evaluation is represented by a scenario tree. This enables us to reformulate the stochastic problem as a large-size mixed-integer linear program. We discuss two alternative formulations, one based on "natural" intuitive inventory level variables, and one based on the echelon stock concept. The echelon stock concept is used to decompose the multi-echelon problem into a series of single-echelon sub-problems linked by coupling constraints.

Chapter 3 is devoted to the resolution of the stochastic problem through a Cut-and-Branch approach. The overall objective of such an approach is to strengthen the linear relaxation of the problem through the use of valid inequalities so as to improve the numerical efficiency of the Branch-and-Bound algorithm embedded in mathematical solvers such as CPLEX. In our work, this is achieved by exploiting valid inequalities proposed by Loparic et al. (2001) for the deterministic single-item single-echelon lot-sizing problem with lost-sales. We first seek to apply these inequalities in the stochastic case by considering only individual scenarios of the scenario tree and a cut generation algorithm is suggested in the section 3.1. Then, we investigate an extension of these valid inequalities in which they are considered for sub-trees (involving several scenarios) of the scenario tree. In view of the complexity of the corresponding separation problem, we consider two alternatives to solve it. Firstly, an exact separation algorithm for sub-trees involving only two scenarios is proposed in the subsection 3.2.2. Secondly, a heuristic separation algorithm based on a local neighborhood search for more general sub-trees is presented in the subsection 3.2.3. Cut generation procedures are included in a Cut-and-Branch algorithm which is described in the section 3.3.

Numerical results aiming at assessing the performance of the proposed Cut-and-Branch solution approaches are presented in the chapter 4. Results show that the new tree cut generation improves the quality of the solution up to 93.89%.

Finally, chapter 5 provides conclusions and discusses potential future works.

Chapter 2

Problem mathematical modeling

Chapter 2 is focused on providing mathematical formulations of the problem described in the previous chapter and is divided into two sections according to the nature of the input data: deterministic (Section 2.1) and stochastic (Section 2.2). In Section 2.1, we introduce a mathematical description of all index and parameters for the problem and provide a mixed-integer linear programming formulation based on a natural 'intuitive' definition of the inventory level decision variables. We then discuss in the Subsection 2.1.2 how our problem relates to other previously studied deterministic lot-sizing problems and study its complexity.

The second part of this chapter is devoted to the mathematical formulation for the stochastic lot-sizing problem. We relax the assumption of deterministically known input data and now assume that the random input parameters, i.e. the demand for remanufactured products, the quantity of used product returned by customers, the proportion of recoverable parts provided by disassembly of used products and the production costs, follow a discrete-time stochastic process. In order to get a computationally tractable model, this stochastic process is approximately represented by a scenario tree (see Section 2.2.1). This approximation allows us to reformulate the stochastic problem as a large-size deterministic mixed-integer linear program (see Subsection 2.2.2). Then, in the Section 2.2.3 we reformulate the previous model by using the echelon stock concept in order to decompose the multi-echelon stochastic problem into single-echelon stochastic sub-problems linked together by inter-echelon coupling constraints.

2.1 Deterministic problem

We first discuss an initial "natural" formulation for this three-echelon uncapacitated lot-sizing problem. We consider a problem involving a single used product and assume that there is an unlimited capacity for the disassembly, refurbishing and reassembly processes. We introduce non-linear production costs, namely fixed setup costs to be incurred each time production takes place in the disassembly, refurbishing or reassembly echelons. We assume that all input parameters needed for production planning decisions are perfectly known at the time the production plan is built.

We first describe our modelling assumptions and introduce the notation for parameters of the problem:

Assumptions:

- There are no capacity restrictions for the disassembling, refurbishing and reassembling processes.

- There might not be enough used products to satisfy the demand: the demand which is not satisfied on time is lost.
- Defective used products and products item are allowed.
- Without loss of generality, the initial inventories (for disassembly, reprocessing, and reassembly) are assumed to be zero.
- The remanufacturing process is considered over a planning horizon with T discrete time periods.

Indices:

- $l = 1, ..., 3$: set of production echelons
  - $l = 1$: disassembly,
  - $l = 2$: refurbishing,
  - $l = 3$: reassembly.
- $p = 1, ..., I + 1$: set of production processes
  - $p = 0$: disassembly,
  - $p = 1...I$: refurbishing of part i,
  - $p = I + 1$: reassembly.
- $i = 0, ..., 2I + 1$: set of products involved in the problem
  - $i = 0$: used product,
  - $i = 1, ..., I$: set of recoverable parts obtained by disassembly of used products,
  - $i = I + 1, ..., 2I$: set of serviceable parts obtained after refurbishing the recoverable parts,
  - $i = 2I + 1$: remanufactured product.
- $t = 1, ..., T$: set of planning periods.

Problem parameters (assumed to be deterministic):

- r_t: number of used products returned at the beginning of period t,
- d_t: demand for remanufactured products to be satisfied at the end of period t,
- α_i: number of parts $i = 1..I$ embedded in one unit of used / remanufactured product. We define $\alpha_0 = \alpha_{2I+1} = 1$ and $\alpha_{i+I} = \alpha_i, \forall i = 1, ..., I$,
- π_{it}: proportion of recoverable parts $i = 1, ..., I$ obtained by disassembling one unit of used product in period $t = 1...T$,
- s_{pt}: setup cost for process p in period t,
- h_{it}: unit inventory holding cost for product $i = 0, ..., 2I + 1$ in period t,
- γ_{it}: unit cost for discarding one unit of product $i = 0, ..., I$,
- l_t: unit penalty cost for losing sales (not satisfying customer demand for re-manufactured product) in period t,
- g_t: unit cost for disassembly process in the period t.

2.1.1 Mixed-integer linear programming formulation

Decision variables:

- X_{pt}: quantity of product processed on process p in period t,
- $Y_{pt} \in \{0,1\}$: setup variable for process p in period t,
- I_{it}: inventory level of product $i = 0, ..., 2I+1$ at the end of period t,
- Q_{it}: quantity of product $i = 0, ..., I$ discarded in period t,
- L_t: lost sales of remanufactured products in period t.

Formulation:

$$Z^* = min \sum_{t=1}^{T} \Big[\sum_{p=0}^{I+1} s_{pt} Y_{pt} + \sum_{i=0}^{2I+1} h_{it} I_{it} + \sum_{i=0}^{I} \gamma_{it} Q_{it} + l_t L_t + g_t X_{0t} \Big] \tag{2.1}$$

$$X_{pt} \leq M_{pt} Y_{pt} \qquad \forall t, \forall p \tag{2.2}$$
$$I_{0,t} = I_{0,t-1} + r_t - X_{0t} - Q_{0t} \qquad \forall t \tag{2.3}$$
$$I_{i,t} = I_{i,t-1} + \pi_{it} \alpha_i X_{0t} - X_{it} - Q_{it} \qquad \forall i = 1..I, \forall t \tag{2.4}$$
$$I_{i,t} = I_{i,t-1} + X_{i-I,t} - \alpha_i X_{I+1,t} \qquad \forall i = I+1..2I, \forall t \tag{2.5}$$
$$I_{2I+1,t} = I_{2I+1,t-1} + X_{I+1,t} - d_t + L_t \qquad \forall t \tag{2.6}$$
$$I_{i,0} = 0 \qquad \forall i = 0..2I+1 \tag{2.7}$$
$$I_{i,t} \geq 0 \qquad \forall i = 0..2I+1, \forall t \tag{2.8}$$
$$L_t \geq 0 \qquad \forall t \tag{2.9}$$
$$X_{pt} \geq 0, Y_{pt} \in \{0,1\} \qquad \forall t, \forall p \tag{2.10}$$
$$Q_{it} \geq 0 \qquad \forall i = 0..I, \forall t \tag{2.11}$$

The objective function (2.1) aims at minimizing the total remanufacturing costs. It comprises the setup costs for all the production processes, the inventory holding costs for all the products involved in the system, the costs for discarding some used products and recoverable parts, the lost sales costs penalizing the non-satisfaction of the customer demand.

Constraints (2.2) ensure that if a production takes place in one of the production processes, the corresponding fixed setup costs are incurred. Note that constraints (2.2) are big-M type constraints. As the production capacity is assumed to be unlimited in the problem, the quantity produced in period t is limited on one hand by the quantity of used products which have been already returned, on the other hand by the total demand which still needs to be satisfied.

- For the process $p = 0$, $M_{0t} = \min \Big(\sum_{\tau=1}^{t} r_\tau, \frac{\sum_{\tau=t}^{T} d_\tau}{\min\limits_{i=1..I} \pi_{it}} \Big)$,
- For the processes $p = 1...I$, $M_{pt} = \min \Big(\sum_{\tau=1}^{t} (\alpha_p r_\tau \max\limits_{\theta=\tau...t} \pi_{p,\theta}), \sum_{\tau=t}^{T} \alpha_p d_\tau \Big)$.
- For the process $p = I+1$, $M_{I+1,t} = min \Big(\sum_{\tau=1}^{t} r_\tau, \sum_{\tau=t}^{T} d_\tau \Big)$. Note that the value of $M_{I+1,t}$ can be further strengthened by replacing $\sum_{\tau=1}^{t} r_\tau$ by $\sum_{\tau=1}^{t} r_\tau (max_{\theta=\tau...t} min_{i=1...I} \pi_{i,\theta})$.

.

Constraints (2.3)-(2.6) are the inventory balance constraints for used products, recoverable and serviceable parts and remanufactured products. Constraint (2.7) defines the initial quantity inventory and constraints (2.8)-(2.11) define characteristics of the decision variables.

Notice that this formulation has been studied by Vu et al. (2017). However, Vu et al. assumed that all parts obtained by disassembling could be recovered (i.e. $\pi_{it} = 1$), discarding used products and recoverable parts were forbidden (i.e. $Q_{it} = 0$) and there was no production cost for disassembling used products (i.e. $g_t = 0$).

2.1.2 Links with other deterministic lot-sizing problems and complexity

Problem (2.1)-(2.11) is related to several variants of the lot-sizing problem which have been studied in the literature. However, it also displays some specific features, which justifies the development of a dedicated solution approach.

- Problem (2.1)-(2.11) corresponds to a lot-sizing problem involving three production echelons (see e.g. Pochet and Wolsey (1991) and Wu et al. (2013)). However, the product structure is a disassembly-reassembly structure which corresponds to none of the series, assembly or general product structures usually investigated in the multi-echelon lot-sizing literature.

- The lot-sizing problem encountered at echelon 1 (disassembly) shares some common features with the lot-sizing variant known as the 'coordinated replenishment problem' and particularly with the 'lot-sizing with strong setup interactions' investigated in Stowers and Palekar (1997) and Bhatia and Palekar (2001). Namely, the authors of Stowers and Palekar (1997) and Bhatia and Palekar (2001) consider a multi-item lot-sizing problem in which there is a joint set-up for a family of items (to be incurred each time one or several items of the family are produced) and the items can only be produced in fixed proportions of one another. However, they focus on a single-echelon production system and consider neither the possibility of discarding products nor of losing sales in their problem.

- Problem (2.1)-(2.11) also involves the management of an inventory of used products fed by deterministic and preset product returns. This has also been considered in several papers dealing with lot-sizing for remanufacturing systems (see e.g. Teunter et al. (2006) and Retel Helmrich et al. (2014)). However these papers assume single-echelon production systems and focus on simultaneously managing manufacturing and remanufacturing activities, rather than on coordinating disassembly and reassembly activities.

- The lot-sizing problem encountered at echelons 1 and 2 is similar to the 'disassembly scheduling' problem investigated e.g. in Kim et al. (2009). This problem focuses on planning the disassembly of used products so as to satisfy a demand on parts but does not consider planning the reassembly operations which will provide remanufactured products.

- Finally, we note that planning production for a remanufacturing system similar to the one considered in the work was investigated by Ahn et al. (2011). However, the authors of Ahn et al. (2011) assume that the quantity of returned products available for disassembly was unlimited and thus incorporate neither inventory balance equations such as (2.3) nor lost sales L_t in their model.

Complexity of the problem

Ahn et al. (2011) considered a particular case of problem (2.1)-(2.11) in which the quantity of returned products is unlimited, the lost sales penalty is assumed to be infinite and discarding used products is forbidden. They proved that in this case, the problem is NP-hard. Thus, problem (2.1)-(2.11) is also NP-hard.

2.2 Stochastic problem

The previous section was devoted to characterize and provide a mathematical representation of the problem is its deterministic version. Now, we focus on the stochastic version of the problem, where among the input parameters of the optimization problem (2.1)-(2.11), some are subject to uncertainty. Throughout the next sections, we will focus on the uncertainties on the quantity of used products returned by users r_t, on the proportion of parts which will be recovered by disassembly, π_{it}, on the demand for remanufactured products d_t, and on the production costs s_{pt}, h_{it}, γ_{it}, g_t. We will consider a multi-stage decision process corresponding to the case where the value of the uncertain parameters unfolds little by little following a discrete-time stochastic process. This leads to the representation of the uncertainty via a scenario tree, in which each decision stage corresponds to a cluster of planning periods. Section 2.2.1 introduces the use of a scenario tree for representing the stochastic information structure, which allows us to model the stochastic problem as multi-stage stochastic integer programming model (2.2.2 and 2.2.3).

2.2.1 Uncertainty representation

We would like to model the situations where a company seeks to plan remanufacturing activities over a multi-period horizon. In each **planning period**, the company has to decide its production and inventory level. In practice, the time is usually discretized into short-length periods (typically corresponding to a shift or a day) in order to ensure the practical relevance of the production plan. However, the time discretization used to update forecasts is usually much sparser (forecasts typically are updated every week or month). Let b be the number of planning periods between each two forecasts updates. New information about uncertainty realization will become available *not* at the end of each period but rather *only after* b periods. Consequently, adjusting the production plan will be required only at the end of every b-periods interval. In other words, the **decision stages** to be used in our multi-stage stochastic approach correspond to sets of b planning periods. Moreover, we assume that the uncertainty realization of all b planning periods belonging to a decision stage s is unfolded at the beginning of the decision stage s.

To reflect the dynamic nature of the decision process, the information structure could be represented by using the scenario tree, where each node n corresponds to a single planning period t belonging to single decision stage s. It represents the state of the world that can be distinguished by the information unfolded up to that period t. At any non-terminal node of the tree, there are one or several branches to indicate future possible outcomes of the random variable from the current node. Each node of time period $t+1$ is thus connected to a single node belonging to time period t. An example of scenario tree with decision stages consisting of multiple periods can be found in Kazemi Zanjani et al. (2010).

We only consider the case where every decision stage has the same *length* (denote **b**) and each node at the last period of the stage (except last stage) has the same *number of immediate children/successors* (denote **c**). Notice that the deterministic problem (2.1)-(2.11) is a particular case of the Stochastic problem, in which $c = 1$, (i.e. one single scenario exists). The Figure 2.1

is the example of a scenario tree for planning over 3 decision stages (3 weeks, new realization of forecast become available at the end of each week), each contains 5 planning periods (5 working days), and each node at the last period of stage 1 and 2 has 3 immediate children ($b = 5, c = 3$).

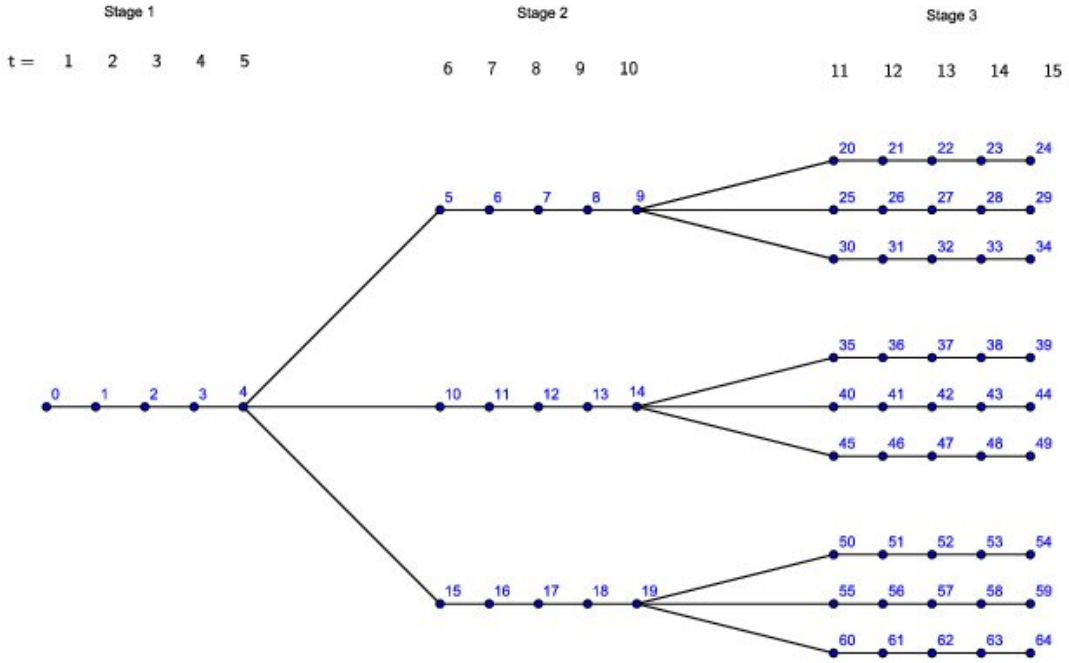

Figure 2.1: Uncertainty representation via a scenario tree

We define the related terms and parameters for the scenario tree:

- $\mathbf{s = 1, 2...S}$: set of decision stages. $\mathbf{t = 1, ...T}$: set of planning periods. Due the same fixed length of stages, $T = b * S$.
- $\mathbf{n = 0...N}$: set of nodes labels in the scenario tree. According to our specific tree structure, we have $N = p\frac{(1-c^S)}{1-c} - 1$. For each node n, $\boldsymbol{s^n}$ denotes the decision stage to which node n belongs. $\boldsymbol{t^n}$ denotes the planning period to which node n belongs.
- $n = 0$ is called *the root node*. Terminal nodes (nodes without any children) are called *leaf nodes*. Each non-terminal node is the root node of a sub-tree $\mathcal{T}(n)$. The set of leaf nodes of sub-tree $\mathcal{T}(n)$ is denoted by $\boldsymbol{\mathcal{L}(n)}$ (we can read $\mathcal{L}(n)$ shortly as *leaf set of node* n and understand it as set of leaf nodes connected to n).
- $\mathbf{a^n}$: *predecessor/parent of node* n *in the tree*. Except for the root node, each node in the tree has one and only one predecessor.
- $\mathcal{P}(\mathbf{n}, \mathbf{m})$: path in the scenario tree between two nodes n and m such that $\tau^n < \tau^m$.
- $\boldsymbol{\rho^{a_n,n}}$: probability transition from node a^n to node n. $\boldsymbol{\rho^{n,m}}$: transition probability from node n to node m. $\rho^{n,m} = \prod_{\nu \in \mathcal{P}(n,m)} \rho^{a(\nu),\nu}$. Then, $\boldsymbol{\rho^n}$ is the occurring probability of node n. We set $\rho^0 = 1$ and for $n \neq 0$, we have: $\rho^n = \rho^{1,n}$; moreover, $\sum_{n,\tau_n=t} \rho^n = 1, \forall t$.

2.2.2 Multi-stage stochastic programming model with full recourse

We propose a multi-stage stochastic programming model corresponding to the uncertainty representation described in the previous section. We consider a full recourse model, i.e. a model in which we assume to have the full flexibility to adjust the production planning at each node of the scenario tree, without taking into account a previously established production plan. The number of parts contained in a finished product is assumed to be deterministic (it does not vary over the nodes). Moreover, we also assume that at each stage, the realization of the random parameters happens before we have to make a decision for this stage, i.e. we assume that we know the exact value of r^n, d^n, l^n, π_i^n, sc_p^n, h_p^n, γ_i^n and g^n when making our production decisions for node n. We describe the notation as follows:

- r^n: quantity of used products (returns) collected at node n,
- d^n: quantity of finished products demanded by customers at node n,
- l^n: unit lost-sales penalty cost,
- sc_0^n: set-up cost for disassembling process at node n. sc_p^n: set-up cost for refurbishing process p at node n, $\forall p = 1, ..., I$. sc_{I+1}^n: set-up cost for reassembling process at node n,
- h_0^n: unit inventory cost for used products at node n; h_i^n: unit inventory cost for part i (recoverable and serviceable parts) at node n, $\forall i = 1, ..., 2I$; h_{2I+1}^n: unit inventory cost for finished products at node n,
- π_i^n: proportion of recoverable parts $i = 1, ..., I$ obtained by disassembling one unit of used product at node n,
- γ_i^n: unit cost for discarding one unit of product $i = 0, ..., I$, at node n,
- g^n: unit cost for disassembly process at node n.

Decision variables:

- X_p^n: quantity of products processed on process $p = 0, ..., I + 1$ at node n,
- $Y_p^n \in \{0, 1\}$: set-up variable for the process p at node n,
- I_i^n: inventory level of product $i = 0, ..., 2I + 1$ when leaving node n,
- Q_i^n: quantity of product $i = 0, ..., I$ discarded at node n,
- L^n: lost sales of remanufactured products at node n.

Multi-stage stochastic integer programming formulation

$$Z^* = min \sum_{n=0}^{N} \rho^n \Big[\sum_{p=0}^{I+1} sc_p^n Y_p^n + \sum_{i=0}^{2I+1} h_i^n I_i^n + \sum_{i=0}^{I} \gamma_i^n Q_i^n + l^n L^n + g^n X_0^n \Big] \tag{2.12}$$

$$X_p^n \leq M_p^n Y_p^n \qquad \forall p, \forall n \tag{2.13}$$

$$I_0^n = I_0^{a^n} + r^n - X_0^n - Q_0^n \qquad \forall n \qquad (2.14)$$

$$I_i^n = I_i^{a^n} + \pi_i^n \alpha_i X_0^n - X_i^n - Q_i^n \qquad \forall i = 1, ..., I, \forall n \qquad (2.15)$$

$$I_i^n = I_i^{a^n} + X_{i-I}^n - \alpha_i X_{I+1}^n \qquad \forall i = I+1, ..., 2I, \forall n \qquad (2.16)$$

$$I_{2I+1}^n = I_{2I+1}^{a^n} + X_{I+1}^n - d^n + L^n \qquad \forall n \qquad (2.17)$$

$$I_0^0 = r_0^0 - X_0^0 - Q_0^0 \qquad (2.18)$$

$$I_i^0 = \pi_i^0 \alpha_i X_0^0 - X_i^0 Q_i^0 \qquad \forall i = 1, ..., I \qquad (2.19)$$

$$I_i^0 = X_{i-I}^0 - \alpha_{i-I} X_{I+1}^0 \qquad \forall i = I+1, ..., 2I \qquad (2.20)$$

$$I_{2I+1} = X_{I+1}^0 - d^0 + l^0 \qquad (2.21)$$

$$I_i^n \geq 0 \qquad \forall i = 0, ..., 2I+1, \forall n \qquad (2.22)$$

$$L^n \geq 0 \qquad \forall n \qquad (2.23)$$

$$X_p^n \geq 0, Y_p^n \in \{0, 1\} \qquad \forall p = 0, ..., I+1, \forall n \qquad (2.24)$$

The objective function (2.25) aims at minimizing the expected cost, over all nodes of the scenario tree. This cost is the sum of the set-up costs, the inventory costs, the lost-sales penalty costs, disposal penalty cost and production cost for disassembly process at each node n multiplied by the occurring probability ρ_n of node n.

Constraints (2.13) link the production quantity variables to the setup variables. The value of the M_p^n constants can be fixed as follows:

- For process $p = 0$, $M_0^n = \min \left\{ \sum_{\nu \in \mathcal{P}(0,n)} r_\nu, \max_{\lambda \in \mathcal{L}(n)} \left\{ \frac{\sum_{\nu \in \mathcal{P}(n,\lambda)} d_\nu}{min_{i=1...I} \pi_i^n} \right\} \right\}$
- For process $p = 1, ..., I$: $M_p^n = \min \left\{ \sum_{\nu \in \mathcal{P}(0,n)} (\alpha_p r_\nu \max_{\mu \in \mathcal{P}(\nu,n)} \pi_p^\mu), \max_{\lambda \in \mathcal{L}(n)} \left\{ \sum_{\nu \in \mathcal{P}(n,\lambda)} \alpha_p d_\nu \right\} \right\}$
- For process $p = I+1$: $M_{I+1}^n = \min \left\{ \sum_{\nu \in \mathcal{P}(0,n)} r_\nu, \max_{\lambda \in \mathcal{L}(n)} \left\{ \sum_{\nu \in \mathcal{P}(n,\lambda)} d_\nu \right\} \right\}$

For each process $p = 0, ..., I+1$ at node n, we take the minimum between the quantity of returned products available to be processed (in the path from root node 0 up to n), and the demand for remanufactured products items needed to be processed (maximum among total demand in the paths from n up to all leaf-nodes in $\mathcal{L}(n)$).

Constraints (2.14)-(2.21) are inventory balance constraints. Note how constraints (2.14) (resp. constraints (2.15) and (2.16)) involve a term corresponding to a dependent demand X_0^n (resp. X_i^n and $\alpha_i X_{I+1}^n$) whereas constraints (2.17) only involve a independent/external demand term d^n. constraints (2.22)-(2.24) characterize the decision variables.

2.2.3 Reformulation of the problem using echelon stock variables

The concept of echelon stock has been widely used to develop solution approaches for multi-echelon lot-sizing problems (see e.g. Pochet and Wolsey (2006)). One of its main advantages is that it helps decomposing the multi-echelon problem into a series of single-echelon lot-sizing problems for which formulation strengthening techniques such as valid inequalities or extended reformulations are available.

Due to the presence of coefficients π_i^n whose value changes over time, it might be quite difficult to define an echelon stock for the used product $i = 0$. We thus focus on defining an echelon stock for products $i = 1, ..., 2I + 1$, which corresponds to a classic assembly structure for multi-echelon lot-sizing problems.

The echelon demand ed_i^n for an intermediate product $i = 1, ..., 2I$ can be understood as the translation of the external demand for the finished product $2I + 1$ into a demand for product i. It can be defined as $ed_i^n = \alpha_i d^n \ \forall i = 1, ..., 2I$.

The echelon stock of a product in a multi-echelon production system corresponds to the total quantity of the product held in inventory in the system, either as such or as component of its successors in the bill-of-material. We introduce the variables $EI_i^n, i = 1, ..., 2I+1$ which correspond to the echelon inventory of product i in the three-echelon production system.

Based on the previous definitions of the echelon demand, we have:

- for the end product $2I + 1$: $EI_{2I+1}^n = I_{2I+1}^n, \forall n$,
- for serviceable parts $i = I + 1, ..., 2I$: $EI_i^n = I_i^n + \alpha_i E_{2I+1}^n = I_i^n + \alpha_i I_{2I+1}^n, \forall n$,
- for recoverable parts $i = 1, ..., I$: $EI_i^n = I_i^n + EI_{I+i}^n = I_i^n + I_{I+i}^n + \alpha_i I_{2I+1}^n, \forall n$

The unit echelon inventory holding cost $eh_i^n, i = 1, ..., 2I + 1$ can be determined by substituting variables I_i^n with variables EI_i^n in the expression computing the total inventory holding cost at node n. Total inventory holding cost and inventory holding cost for each product item are presented as follow:

- $TotIHC = h_0^n I_0^n + \sum_{i=1}^{2I+1} eh_i^n EI_i^n$,
- $eh_i^n = h_i^n, \forall i = 1, ..., I \ \forall n$,
- $eh_i^n = h_i^n - h_{i-I}^n, \forall i = I + 1, ..., 2I; \forall n$,
- $eh_{2I+1}^n = h_{2I+1}^n - \sum_{i=I}^{2I} \alpha_i h_{i+I}^n, \forall n$.

This leads to the following mixed-integer linear programming formulation:

Multi-stage stochastic programming reformulation

$$Z^* = min \sum_{n=0}^{N} \rho^n \Big[\sum_{p=0}^{I+1} sc_p^n Y_p^n + h_0^n I_0^n + \sum_{i=1}^{2I+1} eh_i^n EI_i^n + \sum_{i=0}^{I} \gamma_i^n Q_i^n + l^n L^n + g^n X_0^n \Big] \tag{2.25}$$

$$X_p^n \leq M_p^n Y_p^n \qquad \forall p, \forall n \tag{2.26}$$

$$I_0^n = I_0^{a^n} + r^n - X_0^n - Q_0^n \qquad \forall n \tag{2.27}$$

$$EI_i^n = EI_i^{a^n} + \pi_i^n \alpha_i X_0^n - \alpha_i d^n + \alpha_i L^n - Q_i^n \qquad \forall i = 1..I, \forall n \tag{2.28}$$

$$EI_i^n = EI_i^{a^n} + X_{i-I}^n - \alpha_i d^n + \alpha_i L^n \qquad \forall i = I + 1...2I, \forall n \tag{2.29}$$

$$EI_{2I+1}^n = EI_{2I+1}^{a^n} + X_{I+1}^n - d^n + L^n \qquad \forall n \tag{2.30}$$

$$I_0^0 = r_0^0 - X_0^0 - Q_0^0 \tag{2.31}$$

$$EI_i^0 = \pi_i^0 \alpha_i X_0^0 - \alpha_i d^0 + \alpha_i L^0 - Q_i^0 \qquad \forall i = 1, ..., I \tag{2.32}$$

$$EI_i^0 = X_{i-I}^0 - \alpha_{i-I} d^0 + \alpha_{i-I} L^0 \qquad \forall i = I+1, ..., 2I \tag{2.33}$$

$$EI_{2I+1}^0 = X_{I+1}^0 - d^0 + L^0 \tag{2.34}$$

$$EI_i^n - EI_{I+i}^n \geq 0 \qquad \forall i = 1..I, \forall n \tag{2.35}$$

$$EI_{i,t} - \alpha_i EI_{2I+1,t} \geq 0 \qquad \forall i = I+1...2I, \forall n \tag{2.36}$$

$$EI_{2I+1}^n \geq 0 \qquad \forall n \tag{2.37}$$

$$I_0^n \geq 0 \qquad \forall n \tag{2.38}$$

$$L^n \geq 0 \qquad \forall n \tag{2.39}$$

$$X_p^n \geq 0, Y_p^n \in \{0,1\} \qquad \forall p = 0, ..., I+1, \forall n \tag{2.40}$$

As in the previous formulation, the objective function (2.25) aims at minimizing the expected cost, over all nodes of the scenario tree. This cost is the sum of the set-up costs, the inventory costs, the lost-sales penalty costs, disposal penalty cost and production cost for disassembly process at each node n multiplied to the occurring probability ρ_n of node n. The "big-M" constraints (2.26) are defined as previous formulation.

Constraints (2.27)-(2.31) are inventory balance constraints. Constraints (2.27) make use of the classical inventory wherease constraints (2.28)-(2.30) make use of the echelon inventory. Note how, contrary to constraints (2.15)-(2.16) of the natural formulation, constraints (2.28)-(2.30) do no involve a dependent demand term but only an external demand term. Constraints (2.35)-(2.37) ensure consistency between the echelon inventory at the different levels of the bill-of-material and guarantee that the physical inventory of each product remains non-negative. Finally, constraints (2.39)-(2.40) characterize the decision variables.

Chapter 3

Cut-and-Branch solution approaches for the stochastic problem

Previous chapter was focused on providing a useful mathematical representation for the problem enabling us to apply a polyhedral approach to solve it. Before going on into detailed presentation of this chapter, we recall that we seek to develop an exact solution approach for the problem providing *guaranteed optimal solutions*. *Cut-and-Branch* is the basic algorithm to do that. It carries out an implicit enumeration of all candidate solutions through a search tree (Branch-and-Bound tree) and a lower bound of the optimal solution value is computed (at each node of the tree) in order to discard uninteresting subsets of feasible solutions. In practice, the Branch-and-Bound algorithm performance intensely depends on the quality of this lower bound. The *Branch-and-Cut* algorithm basically is a combination of the Branch-and-Bound algorithm and the idea of utilizing valid inequalities (or cutting planes) to improve the lower bound at the root node of the search tree.

Now, chapter 3 aims at providing a Cut-and-Branch algorithm based on the generation of cutting planes for each sub-problem of the Echelon Stock reformulation in its stochastic version. Chapter 3 will be divided into three sections. Firstly, as mentioned above, Echelon Stock reformulation allows to decompose the problem into a series of single-echelon sub-problem which can be strengthened by generating valid inequalities. In the Section 3.1, we analyze and discuss the reformulation of (k, U) valid inequalities and its corresponding separation algorithms for disassembly and refurbishing/reassembly process. Secondly, we investigate the generation of a new family of valid inequalities, which enables us to improve the linear relaxation of the problem. Section 3.2 introduces the concept of tree inequality and analyze its complicated characteristics. The generation of these valid inequalities implies finding a partition of the nodes in the scenario tree, then we consider two different approaches to generate valid tree inequalities. First, we provide an exact separation approach involving only two different paths of the scenario tree each time. In the Section 3.2.2, we introduce a mixed integer linear programming model to separate each pair of path and define a cut generation algorithm. Second, we consider a heuristic separation approach based on a neighbourhood search involving all paths in the scenario tree, which allows us to define a cut generation algorithm that is presented in the Section 3.2.3. Finally, a Cut-and-Branch algorithm is proposed in the Section 3.3.

3.1 Formulation strengthening by path inequalities

The introduction of echelon inventory variables leads to the elimination of the dependent demand term in the inventory balance equations. This enables us to note that the constraint matrix of (2.25)-(2.40) displays a specific structure: it can be seen as a series of single-echelon single-resource lot-sizing sub-problems coupled by the linking constraints (2.35)-(2.37). These sub-problems are relaxed version of the overall multi-echelon problem. Hence, valid inequalities strengthening the linear relaxation of each of these sub-problems will strengthen the linear relaxation of the overall multi-echelon problem (2.25)-(2.40).

3.1.1 Single-echelon stochastic lot-sizing sub-problems

- Sub-problem for each of the refurbishing/reassembly processes:

For each process $p = 1...I+1$, we have the following sub-problem.

$$Z_p^* = min \sum_{n=0}^{N} \rho^n \left[s_p^n Y_p^n + eh_{p+I}^n EI_{p+I}^n + l^n L^n \right] \tag{3.1}$$

$$X_p^n \leq M_p^n Y_p^n \qquad \forall n \tag{3.2}$$
$$EI_{p+I}^n = EI_{p+I}^{a^n} + X_p^n - \alpha_p d^n + \alpha_p L^n \qquad \forall n \tag{3.3}$$
$$EI_{p+I}^n \geq 0 \qquad \forall n \tag{3.4}$$
$$L^n \geq 0, X_p^n \geq 0, Y_p^n \in \{0,1\} \qquad \forall n \tag{3.5}$$

Sub-problem (3.1)-(3.5) is an uncapacitated single-echelon single-product lot-sizing problem with lost sales. Its deterministic variant was studied among others by Loparic et al. (2001) who proposed a family of valid inequalities to strengthen its linear relaxation. These valid inequalities will be discussed in more detail in the Subsection 3.1 and will be refereed to as (k, U) valid inequalities.

- Sub-problem for the disassembly process:

$$Z_0^* = \min \sum_{n=0}^{N} \rho^n \left[sc_0^n Y_0^n + \sum_{i=1}^{I} eh_i^n EI_i^n + \sum_{i=1}^{I} \gamma_i^n Q_i^n + l^n L^n + g^n X_0^n \right] \tag{3.6}$$

$$X_0^n \leq \min M_0^n Y_0^n \qquad \forall n \tag{3.7}$$
$$EI_i^n = EI_i^{a^n} + \pi_i^n \alpha_i X_0^n - \alpha_i d^n + \alpha_i L^n - Q_i^n \qquad \forall i = 1..I, \forall n \tag{3.8}$$
$$EI_i^n \geq 0 \qquad \forall i = 1..I \tag{3.9}$$
$$L^n \geq 0, X_0^n \geq 0, Y_0^n \in \{0,1\} \qquad \forall n \tag{3.10}$$

Sub-problem (3.6)-(3.10) is an uncapacitated single-echelon multi-product lot-sizing problem with strong setup interactions (see Stowers and Palekar (1997) and Bhatia and Palekar (2001)) and lost sales. Note that the (k, U) inequalities are also valid for the problem (3.6)-(3.10).

3.1.2 Valid path inequalities

A first way of exploiting the (k, U) valid inequalities proposed by Loparic et al. (2001) for the deterministic problem is to consider their application to each individual scenario, i.e. each individual path from the root node to a leaf node, in the scenario tree. This leads to the following expressions:

Let $k \in \mathcal{T}$ be a non-leaf node of the scenario tree. Let $\lambda \in \mathcal{L}(k)$ be a leaf node belonging to the subtree generated by k and let c_k^λ be the immediate successor of the node k belonging to $\mathcal{P}(k, \lambda)$.
Let $U_{k,\lambda} \subset \mathcal{P}(c_k^\lambda, \lambda)$ be a subset of nodes belonging to the individual path between c_k^λ and λ.

For every $p = 1...I + 1$, the following path inequality is valid for sub-problem (3.1)-(3.5).

$$EI_{p+I}^k \geq \alpha_p \sum_{v \in U_{k,\lambda}} \left[d^v \left(1 - \sum_{\mu \in \mathcal{P}(c_k, v)} Y_p^v\right) - L^v \right] \tag{3.11}$$

For each recoverable component $i = 1...I$, the following path inequality is valid for sub-problem (3.6)-(3.10):

$$\min_{i=1...I} \left[\frac{EI_i^k}{\alpha_i} \right] \geq \sum_{v \in U_{k,\lambda}} \left[d^v \left(1 - \sum_{\mu \in \mathcal{P}(c_k, v)} Y_0^v\right) - L^v \right] \tag{3.12}$$

The intuition underlying of path valid inequalities can be understood as follows. We consider the inventory level of the product $p + I$ at the end of node k and look for the future demands for this product in the path defined by the leaf node λ. For a node $v \in U_{k,\lambda}$, if $\sum_{\mu \in \mathcal{P}(c_k, v)} Y_p^v = 1$, the demand of the node v, $\alpha_p d^v$, can be satisfied by a production in one of the node $\mu \in \mathcal{P}(c_k, v)$ and does not have to be in stock at the end of node k. But if $\sum_{\mu \in \mathcal{P}(c_k, v)} Y_p^v \geq 0$, the demand $\alpha_p d^v$ cannot be produced on the interval $\mu \in \mathcal{P}(c_k, v)$: this means that the portion of this demand which will be satisfied by production (i.e. the portion of this demand which will not be lost), $\alpha_p(d^v - L^v)$, should already be in stock at the end of node k.

As the number of these valid inequalities is too large to add all of them a prior in the formulation, Vu et al. (2017) proposed the following cut generation algorithm enabling to add only a subset of the inequalities violated by the current relaxed solution $((\tilde{X}_p, \tilde{Y}_p, \tilde{EI}_{p+I}, \tilde{L})$. The algorithm is defined as follows:

Cut Generation Algorithm

Given a process $p = 0, ..., I + 1$.
For each $k \in \mathcal{T}$, determine subset U_{k,λ^*} using the following rule:
$viol^* = 0$
For each leaf-node $\lambda \in \mathcal{L}(k)$
For each node $v \in \mathcal{P}(c_k^\lambda, \lambda)$
- If

$$d^v \left(1 - \sum_{\mu \in \mathcal{P}(c_k^v, v)} Y_p^v\right) - L^v > 0 \tag{3.13}$$

- **Then** $v \in U_{k,\lambda}$. **Else** $v \notin U_{k,\lambda}$.

-If $p = 1...I+1$:
Compute $viol_\lambda = \tilde{EI}^n_{p+I} - \sum_{v \in U_{k,\lambda}} \alpha_p \big[d^v(1 - \sum_{\mu \in \mathcal{P}(c^v_k, v)} Y^v_p) - L^v\big]$,
- If $p = 0$:
Compute $viol_\lambda = \min\limits_{i=1...I} \left[\frac{\tilde{EI}_{i,k}}{\alpha_i}\right] - \sum_{v \in U_{k,\lambda}} \big[d^v(1 - \sum_{\mu \in \mathcal{P}(c^v_k, v)} Y^v_p) - L^v\big]$,

If $voil_\lambda < viol^*$. Then $viol^* = viol_\lambda$ and $\lambda^* = \lambda$,

- **If** $viol^* < 0$, add the valid inequality corresponding to the leaf node λ, the node k and the process p to the current formulation.
- **Else** there is no violated valid inequality corresponding to node k.

Note that this algorithm seeks to identify the most violated inequality among all inequalities (3.11) or (3.12) available for the process p and the node k.

The numerical results provided by Vu et al. (2017) show that this algorithm leads to a significant decrease in the total computation time needed to obtain guaranteed optimal solutions for medium-size instances of the problem. However, numerical difficulties were encountered for instances where the scenario tree involved more than 1500 nodes. We thus investigate another way of exploiting (k, U) valid inequalities, by considering their application on subtrees involving several scenarios rather than only paths.

3.2 Formulation strengthening by tree inequalities

In the previous section, we provided an adaptation of the valid path inequalities for uncapacitated stochastic lot-sizing problem in remanufacturing systems proposed by Vu et al. (2017) to our more complex problem. In this section, we investigate a new family of valid inequelities by exploiting an idea presented in Guan et al. (2006b) and Guan et al. (2009). They proposed a general scheme to obtain valid inequalities for stochastic integer programs by mixing several valid path inequalities. In what follows, we apply this scheme to derive a new set of tree inequalities based on a mixing of path inequalities discussed in Subsection 3.1. The main challenge here consists in devising an efficient separation algorithm for these new inequalities. This will be discussed in Subsections 3.2.2 and 3.2.3.

3.2.1 Valid tree inequalities

We first recall the relevant notation introduced in the Section 2.2.1. Each node k of the scenario tree $\mathcal{T}$, except for the root node (indexed as $k = 0$), has a unique parent, and each non-terminal node k is the root of a sub-tree $\mathcal{T}(k)$ (notice that $\mathcal{T}(0) := \mathcal{T}$). Let $\mathcal{L}(k)$ be the set of the leaf node connected to the node k in the sub-tree $\mathcal{T}(k)$ and let c^λ_k be the child of node k belonging to $\mathcal{P}(k, \lambda)$, $\lambda \in \mathcal{L}(k)$. Without loss of generality, we drop the index of the production process and assume $\alpha_p = 1$, $\forall\ p = 0, ..., I+1$.

Given a non-leaf node k, we consider $|\mathcal{L}(k)|$ valid path inequalities (one for each leaf node $\lambda \in \mathcal{L}(k)$):

$$EI^k \geq \sum_{\nu \in U_\lambda} \Big[d^\nu \big(1 - \sum_{\mu \in \mathcal{P}(c_k^\lambda, \nu)} Y^\mu\big) - L^\nu \Big]$$

where $U_\lambda \subset \mathcal{P}(c_k^\lambda, \lambda)$ is a subset of nodes belonging to the path between c_k^λ to λ.

By defining $U_\lambda(\mu) = U_\lambda \cap \mathcal{P}(\mu, \lambda)$, we can rewrite these inequalities in a form making it easier to apply Theorem 2 of Guan et al. (2009).

$$EI^k + \sum_{\nu \in U_\lambda} L^\nu + \sum_{\mu \in \mathcal{P}(c_k^\lambda, \lambda)} \big(\sum_{\nu \in U_\lambda(\mu)} d^\nu \big) Y^\mu \geq \sum_{\nu \in U_\lambda} d^\nu$$

We denote $\mathcal{C} = \cup_{\lambda \in \mathcal{L}(k)} \mathcal{P}(c_k^\lambda, \lambda)$ and without loss of generality, we assume that the leaves are indexed in the increasing order of the cumulated demand $\sum_{v \in U_\lambda} d^v$, i.e. $\sum_{\nu \in U_1} d^\nu \leq ... \leq \sum_{\nu \in U_\lambda} d^\nu \leq ... \leq \sum_{\nu \in U_{|\mathcal{L}(k)|}} d^\nu$. Now, we use the Theorem 2 of Guan et al. (2009) to combine these $|\mathcal{L}(k)|$ path inequalities and derive a new tree inequality:

$$EI^k + \sum_{\nu \in \cup_{\lambda \in \mathcal{L}(k)} U_\lambda} L^\nu + \sum_{\mu \in \mathcal{C}} \phi^\mu Y^\mu \geq \max_{\lambda \in \mathcal{L}(k)} \big(\sum_{\nu \in U_\lambda} d^\nu \big)$$

where, for $\mu \in \mathcal{C}$, the coefficient ϕ^μ is given by:

$$\phi^\mu = \min\{ \max_{\lambda \in \mathcal{L}(\mu)} \big(\sum_{\nu \in U_\lambda(\mu)} d^\nu \big), \sum_{\lambda \in \mathcal{L}(\mu)} \big(\sum_{\nu \in U_{\lambda+1}} d^\nu - \sum_{\nu \in U_\lambda} d^\nu \big) \}$$

with $\sum_{v \in U_0} d^v$ set to 0. This leads to the following proposition.

Proposition 1. *Let $k \in \mathcal{T}$ be a non-leaf node of the scenario tree and $U \in \mathcal{T}(k)$ be a subset of nodes belonging to $\mathcal{T}(k)$. For each leaf node $\lambda \in \mathcal{L}(k)$, we denote:*

- *c_k^λ the child of k belonging to $\mathcal{P}(k, \lambda)$,*
- *$U_\lambda = U \cap \mathcal{P}(c_k^\lambda, \lambda)$ and $U_\lambda(\mu) = U \cap \mathcal{P}(\mu, \mu)$*

Let $\sigma_1, ..., \sigma_{|\mathcal{L}(k)|}$ be an ordering of the leaves belonging to $\mathcal{L}(k)$ in the increasing order of the cumulated demand $\sum_{\nu \in U_\lambda} d^\nu$: $\sum_{\nu \in U_{\sigma_0}} d^\nu \leq \sum_{\nu \in U_{\sigma_1}} d^\nu \leq ... \leq \sum_{\nu \in U_{\sigma_l}} d^\nu \leq ... \leq \sum_{\nu \in U_{\sigma_{|\mathcal{L}(k)|}}} d^\nu$. We set $\sum_{\nu \in U_{\sigma_0}} d^\nu = 0$.
Then, the following inequality is valid for problem (2.25)-(2.40):

$$EI^k + \sum_{\nu \in U} L^\nu + \sum_{\mu \in \mathcal{T}(k) \setminus \{k\}} \phi^\mu Y^\mu \geq \sum_{\nu \in U_{\sigma_{|\mathcal{L}(k)|}}} d^\nu \tag{3.14}$$

with $\phi^\mu = \min\{ \max_{\lambda \in \mathcal{L}(\mu)} \big(\sum_{\nu \in U_\lambda(\mu)} d^\nu \big), \sum_{l=1...|\mathcal{L}(k)| \ s.t \ \sigma_l \in \mathcal{L}(\mu)} \big(\sum_{\nu \in U_{\sigma_l}} d^\nu - \sum_{\nu \in U_{\sigma_{l-1}}} d^\nu \big) \}$

Solving the separation problem for inequalities (3.14) for a given non-leaf node k requires identifying which nodes of $\mathcal{T}(k)$ should be selected in the set U in order to minimize the difference

between the left and the right hand side of the inequality. This is challenging as, contrary to the case of path inequalities, it is not possible to consider each node of $\mathcal{T}(k)$ individually. Namely, selecting a node v of $\mathcal{T}(k)$ in the set $\cup_{\lambda\in\mathcal{L}(k)}U_\lambda$ not only changes the left hand side of the inequality by a quantity $L^v + \phi^v Y^v$ but also potentially impacts the value of coefficient ϕ^μ for all other nodes in $\mathcal{T}(k) \setminus \{k\}$ and the value of the right side of the inequality. These interactions significantly complicate the resolution of the separation problem. Thus, we investigate two possible ways of solving it.

3.2.2 Exact separation approach

Subsection 3.2.2 is dedicated to investigating an exact resolution of the separation problem. However, in view of its difficulty, we focus on generating tree valid inequalities on subtrees involving only two leaf nodes.

Let $k \in \mathcal{T}$ be a non-leaf node of the scenario tree and let λ_1 and λ_2 be two leaf nodes belonging to $\mathcal{L}(k)$.

We denote c_k^1 the child of k belonging to $\mathcal{P}(k, \lambda_1)$ and c_k^2 the child of k belonging to $\mathcal{P}(k, \lambda_2)$. As well as the proposition 1 above, we created a valid tree inequality, which involves two path as follows.

We define two path inequalities $U_1 \subset \mathcal{P}(c_k^1, \lambda_1)$, $U_2 \subset \mathcal{P}(c_k^2, \lambda_2)$ and denote $\mathcal{C} = \mathcal{P}(c_k^1, \lambda_1) \cup \mathcal{P}(c_k^2, \lambda_2)$. Applying the Theorem 2 of Guan et al. (2009) we obtain:

$$EI^k + \sum_{\nu\in U_1\cup U_2} L^\nu + \sum_{\mu\in\mathcal{C}} \phi^\mu Y^\mu \geq max\Big(\sum_{\nu\in U_1} d^\nu, \sum_{\nu\in U_2} d^\nu\Big)$$

where the value of ϕ^μ, coefficient of the binary variable Y^μ, depends on the ordering of the path inequalities with respect to the cumulated demand $\sum_{\nu\in U_1} d^\nu \leq \sum_{\nu\in U_2} d^\nu$ or $\sum_{\nu\in U_1} d^\nu \leq \sum_{\nu\in U_2} d^\nu$. In order to ease the computation of ϕ^μ, we divide the set $\mathcal{C}$ into 3 subsets and analyze both cases:

- $\mathcal{C}_0 = \mathcal{P}(c_k^1, \lambda_1) \cap \mathcal{P}(c_k^2, \lambda_2)$ is the set of nodes belonging to both $\mathcal{P}(c_k^1, \lambda_1)$ and $\mathcal{P}(c_k^2, \lambda_2)$,
- $\mathcal{C}_1 = \mathcal{P}(c_k^1, \lambda_1) \setminus \mathcal{C}_0$ is the set of nodes belonging only to $\mathcal{P}(c_k^1, \lambda_1)$,
- $\mathcal{C}_2 = \mathcal{P}(c_k^2, \lambda_2) \setminus \mathcal{C}_0$ is the set of nodes belonging only to $\mathcal{P}(c_k^2, \lambda_2)$.

In case $\sum_{\nu\in U_1} d^\nu \leq \sum_{\nu\in U_2} d^\nu$, we have:

- for nodes $\mu \in \mathcal{C}_0$: $\phi^\mu = min(max(\sum_{\nu\in U_1(\mu)} d^\nu; \sum_{\nu\in U_2(\mu)} d^\nu); \sum_{\nu\in U_2} d^\nu)$
- for nodes $\mu \in \mathcal{C}_1$: $\phi^\mu = min(\sum_{\nu\in U_1(\mu)} d^\nu, \sum_{\nu\in U_1} d^\nu) = \sum_{\nu\in U_1(\mu)} d^\nu$,
- for nodes $\mu \in \mathcal{C}_2$: $\phi^\mu = min(\sum_{\nu\in U_2(\mu)} d^\nu, \sum_{\nu\in U_2} d^\nu - \sum_{\nu\in U_1} d^\nu)$.

In case $\sum_{\nu\in U_1} d^\nu \geq \sum_{\nu\in U_2} d^\nu$, we have:

- for nodes $\mu \in \mathcal{C}_0$: $\phi^\mu = min(max(\sum_{\nu\in U_1(\mu)} d^\nu; \sum_{\nu\in U_2(\mu)} d^\nu); \sum_{\nu\in U_1} d^\nu)$
- for nodes $\mu \in \mathcal{C}_1$: $\phi^\mu = min(\sum_{\nu\in U_1(\mu)} d^\nu, \sum_{\nu\in U_1} d^\nu - \sum_{\nu\in U_2} d^\nu)$,

- for nodes $\mu \in \mathcal{C}_2$: $\phi^\mu = min(\sum_{\nu \in U_2(\mu)} d^\nu, \sum_{\nu \in U_2} d^\nu) = \sum_{\nu \in U_2(\mu)} d^\nu$.

This leads to the following proposition.

Proposition 2. *Let $k \in \mathcal{T}$ be a non-leaf node of the scenario tree and let λ_1 and λ_2 be two leaf nodes belonging to $\mathcal{L}(k)$. We denote $V_0 = V \cap \mathcal{C}_0$, $V_1 = V \cap \mathcal{C}_1$ and $V_2 = V \cap \mathcal{C}_2$. The following inequalities are valid for problem (2.25)-(2.40):*

***case 1**: $\sum_{\nu \in V_1} d^\nu \leq \sum_{\nu \in V_2} d^\nu$*

$$EI^k + \sum_{\nu \in V} L^\nu + \sum_{\mu \in \mathcal{C}} \phi^\mu Y^\mu \geq \sum_{\nu \in V_0 \cup V_2} d^\nu \tag{3.15}$$

with:

- *for $\mu \in \mathcal{C}_0$: $\phi^\mu = \sum_{\nu \in V_0(\mu) \cup V_2} d^\nu$,*
- *for $\mu \in \mathcal{C}_1$: $\phi^\mu = \sum_{\nu \in V_1(\mu)} d^\nu$,*
- *for $\mu \in \mathcal{C}_2$: $\phi^\mu = min\big(\sum_{\nu \in V_2(\mu)} d^\nu; \sum_{\nu \in V_2} d^\nu - \sum_{\nu \in V_1} d^\nu\big)$.*

***case 2**: $\sum_{\nu \in V_1} d^\nu > \sum_{\nu \in V_2} d^\nu$*

$$EI^k + \sum_{\nu \in V} L^\nu + \sum_{\mu \in \mathcal{C}} \phi^\mu Y^\mu \geq \sum_{\nu \in V_0 \cup V_1} d^\nu \tag{3.16}$$

with:

- *for $\mu \in \mathcal{C}_0$: $\phi^\mu = \sum_{\nu \in V_0(\mu) \cup V_1} d^\nu$,*
- *for $\mu \in \mathcal{C}_1$: $\phi^\mu = min\big(\sum_{\nu \in V_1(\mu)} d^\nu; \sum_{\nu \in V_1} d^\nu - \sum_{\nu \in V_2} d^\nu\big)$,*
- *for $\mu \in \mathcal{C}_2$: $\phi^\mu = \sum_{\nu \in V_2(\mu)} d^\nu$.*

Given a solution $(\tilde{EI}, \tilde{X}, \tilde{Y}, \tilde{L})$ of the relaxation problem, we would like to find a violated valid inequality of type (3.15) or (3.16) if it exists. We thus need to identify the set of nodes V which minimizes the difference between the left hand side and the right hand side of inequalities (3.14) and (3.15).

First, we note that we can identify the nodes of $\mathcal{C}_0$ to be added to set V_0 by using the same idea as for the path inequalities.

Namely, note that we have:

$$\sum_{\mu \in \mathcal{C}_0} \phi^\mu Y^\mu = \sum_{\nu \in V_0} d^\nu \Big(\sum_{\mu \in \mathcal{P}(c_k, \nu)} Y^\mu \Big) + \sum_{\mu \in \mathcal{C}_0} max\Big(\sum_{\nu \in V_1} d^\nu, \sum_{\nu \in V_2} d^\nu\Big) Y^\mu$$

This means that inequality (3.14) can be rewritten as:

$$EI^k + \sum_{\nu \in V_0} L^\nu + \sum_{\nu \in V_0} d^\nu \Big(\sum_{\mu \in \mathcal{P}(c_k,\nu)} Y^\mu \Big) + \sum_{\mu \in \mathcal{C}_0} max\Big(\sum_{\nu \in V_1} d^\nu, \sum_{\nu \in V_2} d^\nu \Big) Y^\mu + \sum_{\nu \in V_1 \cup V_2} L^\nu + \sum_{\mu \in \mathcal{C}_1 \cup \mathcal{C}_2} \phi^\mu Y^\mu \geq \sum_{\nu \in V_0} d^\nu + \sum_{\nu \in V_2} d^\nu \tag{3.17}$$

As the value of coefficients ϕ^μ for $\mu \in \mathcal{C}_1 \cup \mathcal{C}_2$ is not impacted by the set of nodes belonging to V_0, we can use the following algorithm to identify the set of nodes belonging to set V_0 in the most violated inequality for each sub-problem. Notice that the same applies to the inequality (3.15).

Separation algorithm for set V_0

Given a non-leaf node $k \in \mathcal{T}$ and a process $i \in [0, I+1]$. Determine subset V_0 using the following rule:

For every process $i = 0, ..., I+1$ and
For $\nu \in \mathcal{C}_0$,
- **If** $\tilde{L}^\nu + d^\nu \big(\sum_{\mu \in \mathcal{P}(c_k,\nu)} \tilde{Y}_i^\mu \big) - d^\nu < 0$, **Then** $\nu \in V_0$,
- **Else** $\nu \notin V_0$.

Now, we need to identify the nodes to be put in sets V_1 and V_2 to form the most violated inequality of type (3.14) and of (3.15). For inequalities of type (3.14), this can be done by solving the following mixed-integer linear program:

Decision variables:

- $\delta^\mu = 1$ if node $\mu \in \mathcal{C}_1 \cup \mathcal{C}_2$ belongs to set V, 0 otherwise
- $\phi^\mu \geq 0$: coefficient of variable $\mu \in \mathcal{C}_2$ in the inequality
- $\epsilon^\mu = 1$ if coefficient of variable $Y^\mu, \mu \in \mathcal{C}_2$ is computed as $\sum_{\nu \in V_2(\mu)} d^\nu$, 0 if it is computed as $\sum_{\nu \in V_2} d^\nu - \sum_{\nu \in V_1} d^\nu$.

Separation model

$$Z^*_{sep} = min \sum_{\mu \in \mathcal{C}_1 \cup \mathcal{C}_2} \tilde{L}^\mu \delta^\mu + \sum_{\mu \in \mathcal{C}_0} \tilde{Y}^\mu \Big(\sum_{\nu \in \mathcal{C}_2} d^\nu \delta^\nu \Big) + \sum_{\mu \in \mathcal{C}_1} \tilde{Y}^\mu \Big(\sum_{\nu \in \mathcal{C}_1(\mu)} d^\nu \delta^\nu \Big) + \sum_{\mu \in \mathcal{C}_2} \tilde{Y}^\mu \phi^\mu - \sum_{\mu \in \mathcal{C}_2} d^\mu \delta^\mu \tag{3.18}$$

$$\sum_{\mu \in \mathcal{C}_1} d^\mu \delta^\mu \leq \sum_{\mu \in \mathcal{C}_2} d^\mu \delta^\mu \tag{3.19}$$

$$\phi^\mu \geq \sum_{\nu \in \mathcal{C}_2(\mu)} d^\nu \delta^\nu - \Big(\sum_{\nu \in \mathcal{C}_2} d^\nu \Big)(1 - \epsilon^\mu) \qquad \forall \mu \in \mathcal{C}_2 \tag{3.20}$$

$$\phi^\mu \geq \sum_{\nu \in \mathcal{C}_2} d^\nu \delta^\nu - \sum_{\nu \in \mathcal{C}_1} d^\nu \delta^\nu - \Big(\sum_{\nu \in \mathcal{C}_2} d^\nu \Big) \epsilon^\mu \qquad \forall \mu \in \mathcal{C}_2 \tag{3.21}$$

$$\delta^{\mu} \in \{0,1\} \qquad \forall \mu \in \mathcal{C}_1 \cup \mathcal{C}_2 \tag{3.22}$$
$$\epsilon^{\mu} \in \{0,1\} \qquad \forall \mu \in \mathcal{C}_2 \tag{3.23}$$

For inequalities of type (3.15), a similar mixed-integer linear program where the role of λ_1 and λ_2 is switched can be solved. The proposed separation algorithm can thus be summarized as follows.

Exact separation algorithm for set V_1 and V_2

Given a non-leaf node $k \in \mathcal{T}$ and a process $i \in [0, ..., I+1]$.

For each pair (λ_1, λ_2) of leaf nodes belonging to $\mathcal{L}(k)$:

1. Build set $V_0 \subset \mathcal{C}_0$ by:
   - Separation algorithm for V_0.

2. Compute $viol_0$ by:
   2a. For each $p = 1, ..., I+1$:
   Compute $viol_0 = \tilde{E}I_{k,i+I} + \alpha_i \sum_{\nu \in V_0} (\tilde{L}^{\nu} + d^{\nu}\big(\sum_{\mu \in \mathcal{P}(c_k,\nu)} \tilde{Y}_i^{\mu}\big) - d^{\nu})$.

   2b. For $p = 0$:
   Compute $viol_0 = \min\limits_{i=1..I} \left[\frac{\tilde{E}I_{k,i}}{\alpha_i}\right] + \sum_{\nu \in V_0} (\tilde{L}^{\nu} + d^{\nu}\big(\sum_{\mu \in \mathcal{P}(c_k,\nu)} \tilde{Y}_0^{\mu}\big) - d^{\nu})$.

3. Solve the MILP (3.18)-(3.23) to build the sets V_1 and V_2.
   - Compute $viol = viol_0 + Z^*_{sep}$.
   - **If** $viol < 0$: we have found the most violated inequality of type (3.14).
   - **Else** $(viol \geq 0)$: there is no violated inequality of (3.14) correponding to nodes k, λ_1 and λ_2.

4. Solve the MILP (3.18)-(3.23) where the role of λ_1 and λ_2 is switched and build the sets V_1 and V_2.
   - Compute $viol = viol_0 + Z^*_{sep}$.
   - **If** $viol < 0$: we have found the most violated inequality of type (3.15).
   - **Else** $viol \geq 0$: there is no violated inequality of (3.15) corresponding to nodes k, λ_1 and λ_2.

Note that this separation algorithm implies solving a large number of mixed integer linear programs. As will be shown by numerical experiments, this leads to prohibitive computation time. Moreover, the algorithm focuses on identifying violated inequalities for subtrees involving only two leaf nodes. Thus, in order to generate more general tree inequalities within a reasonable computation time, we consider in the next section a heuristic resolution of the separation problem.

3.2.3 Heuristic separation approach

The previous subsection was devoted to developing an exact separation approach for the subset of tree inequalities (3.14) obtained by combining only 2 leaf nodes. We now consider the general tree inequalities obtained by combining $|\mathcal{L}(k)|$ path inequalities and develop a heuristic method based on a neighbourhood search to solve the separation problem. Note the algorithm discussed here might

not be able to identify the inequality most violated by the current linear relaxation even if it exists. However, it is intended to find within a limited computational effort some violated inequalities which could contribute in strengthening the formulation. The objective of this subsection is to create an efficient method to explore the performance of the general valid tree inequalities, which cannot be evaluated by the previous subsection.

We recall here that for a given process p and a given node k, solving the separation problem means identifying the subset of nodes $U \subset \mathcal{T}(k) \setminus \{k\}$ so as to minimize the difference between the left-hand side and the right-hand side of the valid inequality (3.14). The purpose of the proposed heuristic method is to identify a subset U providing a violated inequality, even if it is not the most violated one.

As we mention above, the heuristic algorithm is based on a neighbourhood search, i.e. we start with an initial (random or not) set of nodes which is included in the tree inequality and we consider only the possibility to include or discard one node each time. The heuristic method is explained as follows:

Let $\mathcal{S}$ be set of nodes involved in the tree inequality and let $\mathcal{T}(k)$ be the subset of nodes correspoding to the subtree generated by k. In each iteration, the algorithm searches for a neighbour set $S' := \{\mathcal{S} \cup s \text{ or } \mathcal{S} \backslash s' : s \in \mathcal{T}(k) \setminus \mathcal{S} \cup \{k\},\ s' \in \mathcal{S}'\}$ and computes the value of the expression (3.14). Once the algorithm has checked all possible neighbour sets $\mathcal{S}'$, it moves to the best one, i.e. $\mathcal{S} := \mathcal{S}'$ with minimum value of the expression (3.14). Finally, the algorithm stops when there is no neighbour set that has a better value than the set $\mathcal{S}$. Figure 3.1 shows one local iteration of the algorithm representing the set $\mathcal{S}$ by a binary array.

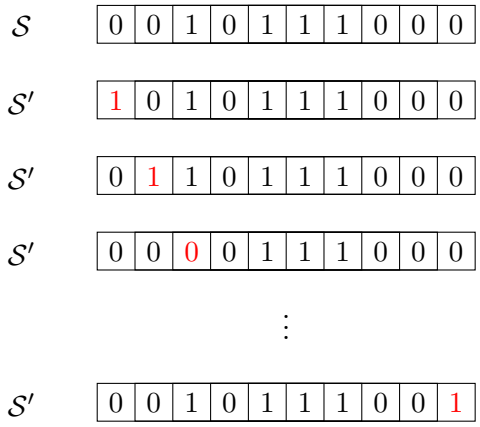

Figure 3.1: One iteration of the heuristic separation algorithm

The heuristic separation algorithm is summarized as follow:

Heuristic separation algorithm

Let $\mathcal{T}' \subseteq \mathcal{T}$ be a subset of nodes which have more than one immediate successor in the scenario tree.

For each node $k \in \mathcal{T}'$

1. Let $\mathcal{M} \subseteq \mathcal{T}(k)$ be a subset of nodes which defines the possible movements for each set $\mathcal{S}'$.
   - Allow each movement $m \in \mathcal{M}$ which satisfy (3.13) for $p = 0, ..., I+1$.
2. Consider an initial (random) partition of the nodes in $T(k)$ and compute

the reference quantity $viol$ given by (3.14)

3. **While** there is still best local movement to be found:

-$viol_c = viol$

While there is still a local movement allowed $m \in \mathcal{M}$

-Move to neighbour solution and compute the quantity $viol_l$ given by (3.14)

-**If** $viol_l < viol_c$. **Then** $viol_c = viol_l$ and save the movement $m' = m$

-Move to the best neighbour solution and $viol = viol_c$.

-Forbid the best local movement m' in the previous iteration, i.e. $\mathcal{M} := \mathcal{M} \backslash \{m'\}$. Note that we forbid the best movement that are used in a local iteration in order to avoid cycling between neighbour sets $\mathcal{S}'$.

4.- **If** $viol < 0$ a new valid tree inequality has been found.

3.3 Cut-and-Branch algorithm

Finally, we come up with a Cut-and-Branch algorithm for solving our stochastic lot-sizing problem. This algorithm relies on the cut generation algorithms proposed in the previous subsections to add valid inequalities to the Echelon Stock formulation discussed in Chapter 2 at the root node of the Branch & Bound tree. The algorithm is summarized as follows:

Cut-and-Branch Algorithm

1. Solve the Linear Relaxation of Echelon Stock Reformulation (LR).
   - **If** the solution is integer, STOP. **Else**, save the solution $(\tilde{X}, \tilde{Y}, \tilde{EI}, \tilde{L})$

2. While there is still some new valid inequality to be found:
   -Initialize a new constraints set $CUTs = \emptyset$.
   For each process $p = 0, ..., I + 1$:
   -Run *Cut generation algorithm* to valid path inequalities
   -Add all found valid inequalities into the set $CUTs$
   -Add all cuts found (i.e. $CUTs$) into the formulation of strengthened (LR).
   -Re-solve this strengthened Linear Relaxation and update new solution into $(\tilde{X}, \tilde{Y}, \tilde{EI}, \tilde{L})$.

3. While there is still some new valid inequality to be found:
   -Initialize a new constraints set $CUTs = \emptyset$.
   For each process $p = 0, ..., I + 1$:
   For each non-leaf node $k \in \mathcal{T}$
   -Run *separation algorithm for set* V_0
   -Run *(Exact or Heuristic) separation algorithm*
   -Add all found valid inequalities into the set $CUTs$
   -Add all cuts found (i.e. $CUTs$) into the formulation of strengthened (LR).
   -Re-solve this strengthened Linear Relaxation and update new solution into $(\tilde{X}, \tilde{Y}, \tilde{EI}, \tilde{L})$.

4. Add integrality constraints and solve this MILP reformulation by Branch-and-Bound algorithm and obtain the optimal solution

Chapter 4

Experimental results

In this chapter, we present some computational results in order to assess the performance of the Cut & Branch algorithm discussed in Chapter 3. Section 4.1 introduces the setting used to randomly generate instances of our problem. Section 4.2 presents the detailed numerical results. All computations have been carried out on the infrastructure of the NLHPC (National Laboratory for High Performance Computing), which consists in computing nodes HP SL230, with Intel Ivy Bridge E5-2660V2 processors. Algorithms are encoded in C++ language and we used ILOG-CPLEX 12.6.1 mathematical programming software package for solving the LPs and MILPs to optimality.

4.1 Instances generation

To compare the performance, instances were randomly generated based on numerical values used by Vu et al. (2017) and Jayaraman (2006) and follow a discrete uniform distribution function with range [a,b]. Generation function for each parameter is presented as follow:

- The lost-sales unit penalty costs l^n were fixed and set to 10000, for all n.
- The demand for finished products d^n in each node were generated from $DU(600, 6000)$.
- The quantity of returned used products r^n in each node were generated from $DU(600, 6000)$.
- The bill of material was generated such that $\alpha_0 = \alpha_I + 1 = 1$, and for each $p = 1, ..., I, \alpha_p$ was generated from $DU(1, 6)$.
- The set-up costs for Disassembly process sc_0^n were generated from $DU(50000, 70000)$, the set-up costs sc_p^n for each Refurbishing processes $p = 1, ..., I$ were generated from $DU(50000, 70000)$, and the set-up cost for the Reassembly process sc_{I+1}^n from $DU(50000, 70000)$.
- The unit inventory holding costs for used products h_0^n was fixed and set to 1. The unit inventory holding costs h_i^n for recoverable parts $i = 1, ..., I$ were generated from $DU(2, 7)$. The serviceable parts unit inventory costs h_i^n were generated from $DU(7, 12)$, $i = I+1, ..., 2I$. Lastly, to ensure non negative echelon costs, we generated finished product unit inventory costs h_{2I+1}^n from $\sum_{i=1}^{I} \alpha_i h_{I+i}^n + \varepsilon$ where we generated ϵ from $DU(50, 100)$.
- Proportion of recoverable parts π_i^n for every $i = 1..I$ obtained by disassembling one unit of used product in the node n are defined by a initial random value for every parts $i = 0, ..., I$ at

$n = 0$, i.e, $\pi_i^n \sim U[0.50, 0.80]$ and for every node $n = 1, ..., N$ we define a normal distribution $\mathcal{N}(0, \sigma)$, where $\sigma \sim U[-0.15, 0.15]$ and $\pi_i^n = \pi_i^0 + \mathcal{N}(0, \sigma)$.

- We define values γ_i^n as proportional value of the inventory cost as follow, $\gamma_i^n = h_i^n * \frac{T}{\beta}$, where $\beta \sim U[2, T]$.
- We define production cost for disassembly process g^n as follows: $g^n = \sum_{i=1}^{I} \gamma_i^n (1 - \pi_i^n)\alpha_i$.

We set the number of components in a product to I = 5, and the length of decision stage to $b = 1$. The number of immediate successors c of each last-period-of-stage node is defined between 3 and 7. As a consequence, the combination of these parameters generated scenario trees which have 43, 85, 121, 156, 259, 341 and 400 nodes. For each scenario tree size, we randomly generated 10 instances. Each of these instances were solved with following solution approaches:

- *Solve echelon stock formulation (EF) directly by ILOG-CPLEX solver*
- *Solve echelon stock formulation (EF) by our Cut-and-Branch algorithm with only valid path inequalities where the auxiliary LPs and strengthened MILP problems are solved by ILOG-CPLEX solver.*
- *Solve echelon stock formulation (EF) by our Cut-and-Branch algorithm with valid path inequalities and valid tree inequalities, which are separated by exact separation algorithm and where the auxiliary LPs and strengthened MILP problems are solved by ILOG-CPLEX solver.*
- *Solve echelon stock formulation (EF) by our Cut-and-Branch algorithm with valid path inequalities and valid tree inequalities, which are separated by heuristic separation algorithm and where the auxiliary LPs and strengthened MILP problems are solved by ILOG-CPLEX solver.*

4.2 Results

We used the default settings of ILOG-CPLEX except for the stopping time criterion, which was set to 900 seconds and the Gomory's cut generation, which is blocked in order to assess performance of our cut generation in the Branch-and-Bound tree. Some of the larger instances could not be solved to optimality before the limit of 900 seconds. For most instances, the quality of (approximated) solution was estimated by the "MIP Gap = ((best integer solution) - Objective value of best node)/(best integer solution)". We also evaluated the quality of the Linear Relaxation with and without (k,U) cuts by the term "LP Gap = (MILP objective value - Linear Relaxation objective value)/MILP objective value". The computational results for I = 5 is given in the Table 4.1.

Results Discussion: *The computational results suggested the optimistic performance of our Cut-and-Branch algorithm, explicitly:*

- Without adding valid inequalities to the Echelon Stock formulation, ILOG-CPLEX had difficulties in solving stochastic problem, even for small instances (43 nodes). Specifically, within the time limit of 900 seconds, ILOG-CPLEX failed to optimize the solutions of Stochastic Echelon Stock formulation for instances with more than 43 nodes.
- Cut-and-Branch algorithm including "*path cut generation*" performed ways better than the direct use of ILOG-CPLEX. Due to the stopping criterion on computation time, both the direct application of CPLEX and the use of our Cut-and-Branch algorithm could not ultimately

				CPLEX Directly on Echelon Stock formulation					
s	c	N	I	LP Gap(%)	MILP Gap (%)	$Cuts^*$	$Time^*$ (s)	B&B Time	Nodes
3	6	43	5	30.55	1.91	-	-	755.49	3863345.80
4	4	85	5	33.27	6.28	-	-	900.09	2303971.10
5	3	121	5	36.21	8.67	-	-	900.06	1714919.70
4	5	156	5	29.92	7.08	-	-	900.13	1384163.80
4	6	259	5	32.49	11.05	-	-	900.12	841860.80
5	4	341	5	34.98	13.14	-	-	900.15	617391.20
4	7	400	5	33.10	11.63	-	-	900.14	546001.86
				Path generation and Cut& Branch on Echelon Stock formulation					
s	c	N	I	LP Gap(%)	MILP Gap (%)	$Cuts^*$	$Time^*$ (s)	B&B Time	Nodes
3	6	43	5	24.23	1.66	103.90	0.04	679.70	3141768.10
4	4	85	5	23.21	5.31	308.60	0.15	900.10	1952797.70
5	3	121	5	21.33	5.32	550.50	0.30	900.02	1221988.70
4	5	156	5	19.81	5.54	515.80	0.42	900.15	1210608.40
4	6	259	5	21.55	8.02	892.00	1.36	900.03	622082.20
5	4	341	5	19.40	7.91	1623.67	2.86	900.02	321483.78
4	7	400	5	19.63	6.95	1449.00	3.58	900.07	373513.00
				Tree exact separation and Cut& Branch on Echelon Stock formulation					
s	c	N	I	LP Gap(%)	MILP Gap (%)	$Cuts^*$	$Time^*$ (s)	B&B Time	Nodes
3	6	43	5	21.79	0.21	2153.90	98.25	238.53	113553.40
4	4	85	5	21.00	1.51	7615.90	616.10	900.22	84743.90
5	3	121	5	19.52	1.84	14631.70	1220.21	900.21	28840.20
4	5	156	5	18.06	2.21	28771.67	2393.93	900.48	43542.11
4	6	259	5	20.74	4.08	81551.10	6378.53	900.35	3050.00
5	4	341	5	19.67	6.97	116684.67	16645.81	900.12	1855.11
4	7	400	5	20.83	7.52	218434.50	12625.13	900.79	3442.75
				Tree heuristic separation and Cut& Branch on Echelon Stock formulation					
s	c	N	I	LP Gap(%)	MILP Gap (%)	$Cuts^*$	$Time^*$ (s)	B&B Time	Nodes
3	6	43	5	17.19	0.12	418.90	1.66	240.95	666866.00
4	4	85	5	17.93	2.06	787.50	7.22	885.31	929840.00
5	3	121	5	16.22	2.50	1180.20	11.08	900.16	355347.00
4	5	156	5	15.21	2.56	1435.60	27.40	900.06	591580.60
4	6	259	5	15.86	3.93	2792.38	148.93	900.47	237287.13
5	4	341	5	15.72	6.07	3465.80	179.97	900.56	83117.70
4	7	400	5	15.29	4.21	4374.57	359.58	900.75	87911.71

Table 4.1: Results obtained by solving each instance using CPLEX and Cut-and-Branch algorithm including valid path inequalities and valid tree inequalities (solving separation problem by an exact separation method and a heuristic method).

optimize the solution within 900 seconds. However, we can observe that after 900 seconds, our algorithm reduced the MILP gap (the difference between the objective function value of best integer solution and the so-far best lower bound) 28.20% on average. Nevertheless, the quality of solutions measured by MILP gap is still large. More specifically, the MILP gap is 5,73% on average.

- Cut-and-Branch algorithm including "*tree cut generation by exact separation algorithm*"

shows a mixed behavior in terms of quality of the optimal solution and computation time. For instances with 85, 121 and 156 nodes present the best MLIP gap between the all the methods. However, in terms of computation time, this method present the worst performance, spending a long time to converge in each set of instances.

- Cut-and-Branch algorithm including "*tree cut generation by heuristic separation algorithm*" present a better performance than the each previous method, i.e. directed use of ILOG-CPLEX, Cut-and-Branch algorithm only including path cut generation and Cut-and-Branch algorithm including tree cut generation by exact separation algorithm. The effectiveness displayed in both computation time and quality of solutions. Our algorithm solved the problems to optimality for small instances (43 nodes), reducing up to 93.89% the MILP gap and significantly reduced the total computation time for instances with up to 85 nodes.

- For medium-size instance, since we set the stopping criterion on computation time, neither algorithm could ultimately optimize the solution within 900 seconds. However, we can observe that after 900 seconds, our Cut-and-Branch algorithm (including tree cut generation) succeeded in closing the MILP gap, reducing the MILP gap 68.90% on average, i.e. the solution approximated by our algorithm (average 2.99%) is much closer to optimality than the approximated solution found by ILOG-CLPEX on its own (average MILP gap is 8.14%).

Regarding to computation time, both exact and heuristic separation algorithm spend a long time to converge for medium-size instances. Unfortunately, this does not allow to compare properly the results above. Hence, we set the time out for the whole Branch & Cut algorithm in 900 seconds and not only for the Branch & Bound part. The new results are presented briefly in the Table 4.2.

		CPLEX on EF		Path B&C on EF		Tree B&C on EF	
N	I	MILP Gap (%)	Nodes	MILP Gap (%)	Nodes	MILP Gap (%)	Nodes
43	5	1.91	3863345.80	1.66	3141768.10	0.12	693301.10
85	5	6.28	2303971.10	5.31	1952797.70	2.32	941936.25
121	5	8.67	1714919.70	5.32	1221988.70	2.50	379284.80
156	5	7.08	1384163.80	5.54	1210608.40	2.58	587158.40
259	5	11.05	841860.80	8.02	622082.20	4.08	221373.50
341	5	13.14	617391.20	7.91	321483.78	6.14	78089.20
400	5	11.63	546001.86	6.95	373513.00	4.58	69328.14

Table 4.2: Results obtained by solving each instance using CPLEX and Cut-and-Branch algorithm including valid path inequalities and valid tree inequalities under the same time out (900 seconds).

Results Discussion: *The computational results suggested that the performance our Cut-and-Branch algorithm is better than CPLEX's performance under same time out (900 seconds), explicitly:*

- Our Cut-and-Branch algorithm not only reduces the MILP gap for each set of instances 66.95% on average, but also reduces the number of travelled nodes to find the integer solution for each set. The number of travelled nodes is reduced 74.99% on average.

Chapter 5

Conclusion

5.1 Findings

- (1) The internship project considers the uncapacitated multi-echelon multi-item stochastic lot-sizing problem in remanufacturing systems with lost sales and uncertain input data. Studying and solving the problem gives theoretical and practical benefits in term of operational research theory, economy and environment protection. We propose models containing three echelons: Disassembling, Refurbishing, and Reassembling; mathematically expressed as multi-stage stochastic integer programming models. Due to the computational experiments, we are confident that our algorithms could be used in real-life situations with small and medium size problems.

- We investigate the stochastic variant of the problem, where input parameters (quantity and quality of returns and demand of customers) are random processes:

  (2) By using the scenario tree concept, we represent the information data structure, and model the stochastic variant as a multi-stage stochastic integer programming model.

  (3) We succeed in modeling a more complex and realistic remanufacturing system, involving uncertain quality of the used products returned by customers.

  (4) We reformulate the problem using the echelon stock concept which enables us to decompose and analyze the corresponding single-echelon sub-problems as well as avoid the strong connection between inventory constraints.

  (5) We succeed in modifying the cuts generation algorithm for sub-problems corresponding to each production process proposed by Vu et al. (2017).

  (6) We succeed in applying and writing the specific form of valid tree inequalities class for our sub-problems. We prove the validity and effectiveness of tree inequalities for our problem.

  (7) We develop a new solution approach to deal with separation problem of generating valid tree inequalities and we succeed in developing a heuristic resolution method to solve the corresponding separation problem.

  (8) We design a Cut-and-Branch algorithm on the stochastic variant of the problem under echelon stock reformulation, and then we run some computational experiments to test its performance. The results show the significant improvements comparing to algorithms implemented by ILOG-CPLEX solver itself. Our Cut-and-Branch algorithm succeeds in solving to optimality small-size problems. In general, it successively reduces the computation time and enhances the quality of solutions on computational experiments .

5.2 Project Limitation and Extension

In this present work, we keep assumption of an unlimited production capacity (only with set-up cost), it would be interesting to see how the algorithms behave in case the production capacity is limited. Moreover, in real-life situations, not only the demand and returns quantity/quality are uncertain, but also parts with different conditions of quality may require different amounts and types of processes in order to be restored (stochastic routings). This will thus have impact on the data structure of the problems. For the time being, we have not been able to effectively reflect this stochastic factor of stochastic routing in our model, since it is not a straightforward application as demand and returns quantity/quality. On the other hand, we consider a single-product problem. Relaxing this assumption and moving to multi-product problem is a next step to be closer to the real-life remanufacturing systems.

Regarding to resolution approaches for the separation problem, we cannot solve large-size instances in a reasonable computation time. Hence, investigate another solution approaches (exact and heuristic) in order to reduce the total computation time should be a next step. In constructing the scenario tree, as well as previous works, we made some assumptions on the probability transition, the length of stages, the number of immediate children, etc. Relaxing these assumptions is another potential perspective for future work.

Bibliography

Nabil Absi and Safia Kedad-Sidhoum. The multi-item capacitated lot-sizing problem with setup times and shortage costs. *European journal of operational research*, 185(3):1351–1374, 2008.

Hyung-Dae Ahn, Dong-Ho Lee, and Hwa-Joong Kim. Solution algorithms for dynamic lot-sizing in remanufacturing systems. *International Journal of Production Research*, 49(22):6729–6748, 2011.

Patrizia Beraldi, Gianpaolo Ghiani, Antonio Grieco, and Emanuela Guerriero. Fix and relax heuristic for a stochastic lot-sizing problem. *Computational Optimization and Applications*, 33(2-3): 303–318, 2006.

Manish Bhatia and Udatta S Palekar. A variable redefinition approach for the lot sizing problem with strong set-up interactions. *Iie Transactions*, 33(5):357–370, 2001.

James H Bookbinder and Jin-Yan Tan. Strategies for the probabilistic lot-sizing problem with service-level constraints. *Management Science*, 34(9):1096–1108, 1988.

Paolo Brandimarte. Multi-item capacitated lot-sizing with demand uncertainty. *International Journal of Production Research*, 44(15):2997–3022, 2006.

Marco Di Summa and Laurence A Wolsey. Lot-sizing on a tree. *Operations Research Letters*, 36 (1):7–13, 2008.

Moritz Fleischmann, Jacqueline M Bloemhof-Ruwaard, Rommert Dekker, Erwin Van der Laan, Jo AEE Van Nunen, and Luk N Van Wassenhove. Quantitative models for reverse logistics: A review. *European journal of operational research*, 103(1):1–17, 1997.

Yongpei Guan and Andrew J Miller. Polynomial-time algorithms for stochastic uncapacitated lot-sizing problems. *Operations Research*, 56(5):1172–1183, 2008.

Yongpei Guan, Shabbir Ahmed, Andrew J Miller, and George L Nemhauser. On formulations of the stochastic uncapacitated lot-sizing problem. *Operations Research Letters*, 34(3):241–250, 2006a.

Yongpei Guan, Shabbir Ahmed, George L Nemhauser, and Andrew J Miller. A branch-and-cut algorithm for the stochastic uncapacitated lot-sizing problem. *Mathematical Programming*, 105 (1):55–84, 2006b.

Yongpei Guan, Shabbir Ahmed, and George L Nemhauser. Sequential pairing of mixed integer inequalities. *Discrete Optimization*, 4(1):21–39, 2007.

Yongpei Guan, Shabbir Ahmed, and George L Nemhauser. Cutting planes for multistage stochastic integer programs. *Operations research*, 57(2):287–298, 2009.

V Daniel R Guide. Production planning and control for remanufacturing: industry practice and research needs. *Journal of operations Management*, 18(4):467–483, 2000.

V Daniel R Guide, Vaidyanathan Jayaraman, and Rajesh Srivastava. Production planning and control for remanufacturing: a state-of-the-art survey. *Robotics and Computer-Integrated Manufacturing*, 15(3):221–230, 1999.

Askiner Gungor and Surendra M Gupta. Issues in environmentally conscious manufacturing and product recovery: a survey. *Computers & Industrial Engineering*, 36(4):811–853, 1999.

Vesra Hashemi, Mingyuan Chen, and Liping Fang. Process planning for closed-loop aerospace manufacturing supply chain and environmental impact reduction. *Computers & Industrial Engineering*, 75:87–95, 2014.

Kjetil K Haugen, Arne Løkketangen, and David L Woodruff. Progressive hedging as a meta-heuristic applied to stochastic lot-sizing. *European Journal of Operational Research*, 132(1):116–122, 2001.

Kai Huang and Simge KüçüKyavuz. On stochastic lot-sizing problems with random lead times. *Operations Research Letters*, 36(3):303–308, 2008.

Hark-Chin Hwang, Wilco van den Heuvel, and Albert PM Wagelmans. The economic lot-sizing problem with lost sales and bounded inventory. *IIE Transactions*, 45(8):912–924, 2013.

Mehmet Ali Ilgin and Surendra M Gupta. Environmentally conscious manufacturing and product recovery (ecmpro): a review of the state of the art. *Journal of environmental management*, 91 (3):563–591, 2010.

Raf Jans and Zeger Degraeve. Meta-heuristics for dynamic lot sizing: A review and comparison of solution approaches. *European Journal of Operational Research*, 177(3):1855–1875, 2007.

Raf Jans and Zeger Degraeve. Modeling industrial lot sizing problems: a review. *International Journal of Production Research*, 46(6):1619–1643, 2008.

Vaidyanathan Jayaraman. Production planning for closed-loop supply chains with product recovery and reuse: an analytical approach. *International Journal of Production Research*, 44(5):981–998, 2006.

Muris Lage Junior and Moacir Godinho Filho. Production planning and control for remanufacturing: literature review and analysis. *Production Planning & Control*, 23(6):419–435, 2012.

Muris Lage Junior and Moacir Godinho Filho. Master disassembly scheduling in a remanufacturing system with stochastic routings. *Central European Journal of Operations Research*, 25(1):123–138, 2017.

Kyung-Wan Kang, Hyoung-Ho Doh, Jung-Hyeon Park, and Dong-Ho Lee. An integrated model and its extension for disassembly leveling and lot-sizing for multiple product types. In *System Integration (SII), 2011 IEEE/SICE International Symposium on*, pages 756–761. IEEE, 2011.

Masoumeh Kazemi Zanjani, Mustapha Nourelfath, and Daoud Ait-Kadi. A multi-stage stochastic programming approach for production planning with uncertainty in the quality of raw materials and demand. *International Journal of Production Research*, 48(16):4701–4723, 2010.

HJ Kim, Dong-Ho Lee, P Xirouchakis, and OK Kwon. A branch and bound algorithm for disassembly scheduling with assembly product structure. *Journal of the Operational Research Society*, 60(3):419–430, 2009.

Kibum Kim, Iksoo Song, Juyong Kim, and Bongju Jeong. Supply planning model for remanufacturing system in reverse logistics environment. *Computers & Industrial Engineering*, 51(2): 279–287, 2006.

Simge Küçükyavuz. On mixing sets arising in chance-constrained programming. *Mathematical programming*, 132(1):31–56, 2012.

Muris Lage Junior and Moacir Godinho Filho. Production planning and control for remanufacturing: exploring characteristics and difficulties with case studies. *Production Planning & Control*, 27(3):212–225, 2016.

Jianzhi Li, Miguel González, and Yun Zhu. A hybrid simulation optimization method for production planning of dedicated remanufacturing. *International Journal of Production Economics*, 117(2): 286–301, 2009.

Yongjian Li, Jian Chen, and Xiaoqiang Cai. Uncapacitated production planning with multiple product types, returned product remanufacturing, and demand substitution. *OR Spectrum*, 28 (1):101–125, 2006.

Marko Loparic, Yves Pochet, and Laurence A Wolsey. The uncapacitated lot-sizing problem with sales and safety stocks. *Mathematical Programming*, 89(3):487–504, 2001.

Robert T Lund. Remanufacturing. *Technology review*, 87(2):18, 1984.

Pedro Belluco Macedo, Douglas Alem, Maristela Santos, Muris Lage Junior, and Alfredo Moreno. Hybrid manufacturing and remanufacturing lot-sizing problem with stochastic demand, return, and setup costs. *The International Journal of Advanced Manufacturing Technology*, 82(5-8): 1241–1257, 2016.

Mohd Arshad Naeem, Dean J Dias, Rupak Tibrewal, Pei-Chann Chang, and Manoj Kumar Tiwari. Production planning optimization for manufacturing and remanufacturing system in stochastic environment. *Journal of Intelligent Manufacturing*, pages 1–12, 2013.

Pedro Pineyro and Omar Viera. The economic lot-sizing problem with remanufacturing and one-way substitution. *International Journal of Production Economics*, 124(2):482–488, 2010.

Grigoris S Piperagkas, I Konstantaras, K Skouri, and Konstantinos E Parsopoulos. Solving the stochastic dynamic lot-sizing problem through nature-inspired heuristics. *Computers & Operations Research*, 39(7):1555–1565, 2012.

Yves Pochet and Laurence A Wolsey. Solving multi-item lot-sizing problems using strong cutting planes. *Management Science*, 37(1):53–67, 1991.

Yves Pochet and Laurence A Wolsey. *Production planning by mixed integer programming*. Springer Science & Business Media, 2006.

Mathijn J Retel Helmrich, Raf Jans, Wilco van den Heuvel, and Albert PM Wagelmans. Economic lot-sizing with remanufacturing: complexity and efficient formulations. *IIE Transactions*, 46(1): 67–86, 2014.

Florian Sahling. A column-generation approach for a short-term production planning problem in closed-loop supply chains. 2013.

Curtis L Stowers and Udatta S Palekar. Lot sizing problems with strong set-up interactions. *IIE transactions*, 29(2):167–179, 1997.

Jinmo Sung and Bongju Jeong. A heuristic for disassembly planning in remanufacturing system. *The Scientific World Journal*, 2014, 2014.

Lixin Tang, Ping Che, and Jiyin Liu. A stochastic production planning problem with nonlinear cost. *Computers & Operations Research*, 39(9):1977–1987, 2012.

S Armagan Tarim and Brian G Kingsman. The stochastic dynamic production/inventory lot-sizing problem with service-level constraints. *International Journal of Production Economics*, 88(1): 105–119, 2004.

Horst Tempelmeier. On the stochastic uncapacitated dynamic single-item lotsizing problem with service level constraints. *European Journal of Operational Research*, 181(1):184–194, 2007.

Horst Tempelmeier and Sascha Herpers. Dynamic uncapacitated lot sizing with random demand under a fillrate constraint. *European Journal of Operational Research*, 212(3):497–507, 2011.

Ruud H Teunter, Z Pelin Bayindir, and Wilco Van Den Heuvel. Dynamic lot sizing with product returns and remanufacturing. *International Journal of Production Research*, 44(20):4377–4400, 2006.

Vicente Vargas. An optimal solution for the stochastic version of the wagner–whitin dynamic lot-size model. *European Journal of Operational Research*, 198(2):447–451, 2009.

Quan Dong Vu, Céline Gicquel, and Safia Kedad-Sidhoum. Stochastic programming approaches for planning re-manufacturing activities under uncertain demand and returns forecasts. In *ROADEF 2017-18e conférence de la société française de Recherche Opérationnelle et d'Aide à la décision*, 2017.

Harvey M Wagner and Thomson M Whitin. Dynamic version of the economic lot size model. *Management science*, 5(1):89–96, 1958.

Hsiao-Fan Wang and Yen-Shan Huang. A two-stage robust programming approach to demand-driven disassembly planning for a closed-loop supply chain system. *International Journal of Production Research*, 51(8):2414–2432, 2013.

Tao Wu, Canrong Zhang, Zhe Liang, and Stephen CH Leung. A lagrangian relaxation-based method and models evaluation for multi-level lot sizing problems with backorders. *Computers & Operations Research*, 40(7):1852–1863, 2013.

Masoumeh Kazemi Zanjani, Daoud Ait-Kadi, and Mustapha Nourelfath. An accelerated scenario updating heuristic for stochastic production planning with set-up constraints in sawmills. *International Journal of Production Research*, 51(4):993–1005, 2013.

Minjiao Zhang, Simge Küçükyavuz, and Hande Yaman. A polyhedral study of multiechelon lot sizing with intermediate demands. *Operations Research*, 60(4):918–935, 2012.

Minjiao Zhang, Simge Küçükyavuz, and Saumya Goel. A branch-and-cut method for dynamic decision making under joint chance constraints. *Management Science*, 60(5):1317–1333, 2014.

Chaoyue Zhao and Yongpei Guan. Extended formulations for stochastic lot-sizing problems. *Operations Research Letters*, 42(4):278–283, 2014.