A stochastic dual dynamic integer programming based approach for remanufacturing planning under uncertainty

Franco Quezada, Céline Gicquel, Safia Kedad-Sidhoum

A stochastic dual dynamic integer programming based approach for remanufacturing planning under uncertainty

Franco Quezada[a,b], Céline Gicquel[c] and Safia Kedad-Sidhoum[d]

[a]University of Santiago of Chile (USACH), Faculty of Engineering, Industrial Engineering Department, Chile;
[b]University of Santiago of Chile (USACH), Faculty of Engineering, Program for the Development of Sustainable Production Systems (PDSPS), Chile;
[c]Université Paris Saclay, LISN, 91190 Gif-sur-Yvette, France;
[d]CNAM, CEDRIC, 75003 Paris, France.

ARTICLE HISTORY
Compiled September 20, 2022

Abstract
We seek to optimize the production planning of a three-echelon remanufacturing system under uncertain input data. We consider a multi-stage stochastic integer programming approach and use scenario trees to represent the uncertain information structure. We introduce a new dynamic programming formulation that relies on a partial nested decomposition of the scenario tree. We then propose a new approximate stochastic dual dynamic integer programming algorithm based on this partial decomposition. Our numerical results show that the proposed solution approach is able to provide near-optimal solutions for large-size instances with a reasonable computational effort.

KEYWORDS
Production planning ; Lot-sizing ; Remanufacturing ; Multi-stage stochastic integer programming; Stochastic dual dynamic programming.

1. Introduction

Nowadays, linear open-ended production systems (i.e., produce, consume, dispose) are facing significant challenges in terms of design, operation, and resource management. This is mainly due to the advent of environmental regulations around the world, which force industrial companies to become more environmentally responsible and to mitigate the environmental impact of their products (Suzanne, Absi, and Borodin 2020). In contrast, circular closed-loop production systems aim at achieving more sustainability in industrial operations through, e.g., reuse of products in a sufficiently good working condition, remanufacturing of end-of-life products and recycling of raw materials.

This work considers one aspect of closed-loop production systems: the remanufacturing of end-of-life products. Remanufacturing is defined as a set of processes transforming end-of-life products (also called used products or returns) into like-new finished products, once again usable by customers, mainly by rehabilitating damaged

*Corresponding author. Email: franco.quezada@usach.cl
*Preliminary results were published in Quezada, Gicquel, and Kedad-Sidhoum (2021b).

components (Lund 1984). Remanufacturing makes it possible to reuse the components embedded in used products without fully dismantling them into basic raw materials. It thus prevents pollution emissions and natural resource consumption and saves the energy and resources needed to transform basic raw materials into components. The fact that remanufacturing is of particular interest for a wide range of industries can be seen through the numerous case studies reported in the academic literature. Industrial systems aimed at remanufacturing cars (Saavedra et al. 2013; Xiang and Ming 2011), photocopiers (Kerr and Ryan 2001), mobile handsets (Rathore, Kota, and Chakrabarti 2011), electrical and electronic equipment (Hatcher, Ijomah, and Windmill 2013) or office furniture (Krystofik et al. 2018) were thus recently studied.

We focus here on an optimization problem related to the operational management of remanufacturing systems, namely short-term production planning and lot-sizing. In general, short-term production planning involves determining the production level (i.e., which products and how much of them should be made), the timing (i.e., when the products should be made) and the resources to be used over a multi-period horizon. Within the remanufacturing context investigated here, production planning includes making decisions on how much and when used products should be disassembled, refurbished or reassembled. The goal is to meet the customers' demand for remanufactured products in the most cost-effective way. In many cases, the costs to be taken into account when planning production involve setup costs. This situation occurs when setup operations such as tool changes or machine calibration must be undergone on the production resources before starting the production of a new type of product. Setup costs are usually fixed costs, i.e., costs whose value does not depend on the amount of product produced after the setup. In this case, one may wish to plan production using large lot sizes to benefit from economies of scale. But this leads to a desynchronization between the production plan and the demand for the finished products, and consequently to high levels of costly inventory. Lot-sizing models aim at building production plans in which the trade-off between fixed setup costs and inventory holding costs is optimized, and the customers' demand is satisfied as best as possible. In their recent state of the art on discrete-time tactical production planning models for the circular economy, Suzanne, Absi, and Borodin (2020) found that lot-sizing models for remanufacturing systems involving both disassembly and reassembly operations have not been fully investigated, although such systems are frequently encountered in practice. This work may be seen as a way of partially closing this gap. We thus seek to optimize short-term production planning and lot-sizing for a production system comprising three production echelons: the disassembly of used products brought back by customers into parts, the refurbishing of the recovered parts, and the reassembly of refurbished parts into like-new finished products.

Furthermore, the fact that production planning is more complex in remanufacturing systems than in classical manufacturing systems is now well established in the literature (see, e.g., Guide, Jayaraman, and Srivastava (1999) and Guide (2000)). This difficulty comes partly from the fact that the remanufacturer has a lower control on the input flows of used products returned by customers, both in terms of quantity and quality, than the one a manufacturer has on the input flows of raw materials and components ordered from its suppliers. This means that the input data needed to make planning decisions are subject to a higher level of uncertainty in remanufacturing systems than in classical manufacturing systems. Neglecting these uncertainties while planning production may result in production plans displaying a poor performance in practice. We thus investigate a production planning and lot-sizing model in which the uncertainties related to the quantity and quality of returned products, the customers' demand, and

the costs are explicitly taken into account. However, we assume that even if these input parameters are subject to uncertainties, some knowledge about their potential value is available under the form of probability distributions. Moreover, production planning is most often a multi-stage decision-making process. Namely, part of the production decisions must be made 'here and now', i.e., right at the beginning of the planning horizon, but others may be postponed until more information on the stochastic input parameters becomes available using a 'wait-and-see' strategy. In order to explicitly consider this multi-stage aspect, we use a multi-stage stochastic integer programming (MSSiP) model in which the time evolution of the uncertain parameters is described through a discrete scenario tree.

Le suivi des modifications est activé A stochastic dual dynamic integer programming based approach for remanufacturing planning under uncertainty Franco Quezadaa,b, Céline Gicquelc and Safia Kedad-Sidhoumd aUniversity of Santiago of Chile (USACH), Faculty of Engineering, Industrial Engineering Department, Chile; bUniversity of Santiago of Chile (USACH), Faculty of Engineering, Program for the Development of Sustainable Production Systems (PDSPS), Chile; cUniversité Paris Saclay, LISN, 91190 Gif-sur-Yvette, France; dCNAM, CEDRIC, 75003 Paris, France. ARTICLE HISTORY Compiled September 20, 2022 Abstract We seek to optimize the production planning of a three-echelon remanufacturing system under uncertain input data. We consider a multi-stage stochastic integer programming approach and use scenario trees to represent the uncertain informa- tion structure. We introduce a new dynamic programming formulation that relies on a partial nested decomposition of the scenario tree. We then propose a new approx- imate stochastic dual dynamic integer programming algorithm based on this partial decomposition. Our numerical results show that the proposed solution approach is able to provide near-optimal solutions for large-size instances with a reasonable computational effort. KEYWORDS Production planning ; Lot-sizing ; Remanufacturing ; Multi-stage stochastic integer programming; Stochastic dual dynamic programming. 1. Introduction Nowadays, linear open-ended production systems (i.e., produce, consume, dispose) are facing significant challenges in terms of design, operation, and resource management. This is mainly due to the advent of environmental regulations around the world, which force industrial companies to become more environmentally responsible and to mitigate the environmental impact of their products (Suzanne, Absi, and Borodin 2020). In contrast, circular closed-loop production systems aim at achieving more sustainability in industrial operations through, e.g., reuse of products in a sufficiently good working condition, remanufacturing of end-of-life products and recycling of raw materials. This work considers one aspect of closed-loop production systems: the remanufac- turing of end-of-life products. Remanufacturing is defined as a set of processes trans- forming end-of-life products (also called used products or returns) into like-new fin- ished products, once again usable by customers, mainly by rehabilitating damaged *Corresponding author. Email: franco.quezada@usach.cl *Preliminary results were published in Quezada, Gicquel, and Kedad-Sidhoum (2021b). components (Lund 1984). Remanufacturing makes it possible to reuse the components embedded in used products without fully dismantling them into basic raw materials. It thus prevents pollution emissions and natural resource consumption and saves the energy and resources needed to transform basic raw materials into components. The fact that remanufacturing is of particular interest for a wide range of industries can be seen through the numerous case studies reported in the academic literature. Industrial systems aimed at remanufacturing cars (Saavedra et al. 2013; Xiang and Ming 2011), photocopiers (Kerr and Ryan 2001), mobile handsets (Rathore, Kota, and Chakrabarti 2011), electrical and electronic equipment (Hatcher, Ijomah, and Windmill 2013) or

office furniture (Krystofik et al. 2018) were thus recently studied. We focus here on an optimization problem related to the operational management of remanufacturing systems, namely short-term production planning and lot-sizing. In general, short-term production planning involves determining the production level (i.e., which products and how much of them should be made), the timing (i.e., when the products should be made) and the resources to be used over a multi-period horizon. Within the remanufacturing context investigated here, production planning includes making decisions on how much and when used products should be disassembled, refur- bished or reassembled. The goal is to meet the customers' demand for remanufactured products in the most cost-effective way. In many cases, the costs to be taken into ac- count when planning production involve setup costs. This situation occurs when setup operations such as tool changes or machine calibration must be undergone on the pro- duction resources before starting the production of a new type of product. Setup costs are usually fixed costs, i.e., costs whose value does not depend on the amount of product produced after the setup. In this case, one may wish to plan production using large lot sizes to benefit from economies of scale. But this leads to a desynchronization between the production plan and the demand for the finished products, and consequently to high levels of costly inventory. Lot-sizing models aim at building production plans in which the trade-off between fixed setup costs and inventory holding costs is optimized, and the customers' demand is satisfied as best as possible. In their recent state of the art on discrete-time tactical production planning models for the circular economy, Suzanne, Absi, and Borodin (2020) found that lot-sizing models for remanufacturing systems involving both disassembly and reassembly operations have not been fully investigated, although such systems are frequently encountered in practice. This work may be seen as a way of partially closing this gap. We thus seek to optimize short-term production planning and lot-sizing for a production system comprising three production echelons: the disassembly of used products brought back by customers into parts, the refurbishing of the recovered parts, and the reassembly of refurbished parts into like-new finished products. Furthermore, the fact that production planning is more complex in remanufacturing systems than in classical manufacturing systems is now well established in the literature (see, e.g., Guide, Jayaraman, and Srivastava (1999) and Guide (2000)). This difficulty comes partly from the fact that the remanufacturer has a lower control on the input flows of used products returned by customers, both in terms of quantity and quality, than the one a manufacturer has on the input flows of raw materials and components ordered from its suppliers. This means that the input data needed to make planning decisions are subject to a higher level of uncertainty in remanufacturing systems than in classical manufacturing systems. Neglecting these uncertainties while planning pro- duction may result in production plans displaying a poor performance in practice. We thus investigate a production planning and lot-sizing model in which the uncertainties related to the quantity and quality of returned products, the customers' demand, and 2 the costs are explicitly taken into account. However, we assume that even if these input parameters are subject to uncertainties, some knowledge about their potential value is available under the form of probability distributions. Moreover, production planning is most often a multi-stage decision-making process. Namely, part of the production decisions must be made 'here and now', i.e., right at the beginning of the planning horizon, but others may be postponed until more information on the stochastic input parameters becomes available using a 'wait-and-see' strategy. In order to explicitly consider this multi-stage aspect, we use a multi-stage stochastic integer programming (MSSiP) model in which the time evolution of the uncertain parameters is described through a discrete scenario

tree. The problem studied here, i.e., stochastic production planning and lot-sizing for a three-echelon remanufacturing system, was previously investigated by Quezada, Gicquel, and Kedad-Sidhoum (2019a) and Quezada et al. (2020). Quezada et al. (2020) formulated the problem as a large-size mixed-integer linear program (MILP) and proposed a customized branch-and-cut algorithm based on new valid inequalities to solve it. Quezada, Gicquel, and Kedad-Sidhoum (2019a) later investigated the use of the Stochastic Dual Dynamic integer Programming (SDDiP) algorithm recently presented by Zou, Ahmed, and Sun (2019) to solve the problem. This algorithm relies on a full decomposition of the problem into a large number of small deterministic sub-problems. Although the above-mentioned solution approaches successfully provided near-optimal solutions for small to medium-size instances, some numerical difficulties were encountered to solve instances involving the large-size scenario trees which are often needed in real applications. The present work thus discusses a new solution approach for this problem. This one uses a partial decomposition of the problem into a small number of medium-size stochastic sub-problems. Note that this partial decomposition was successfully imple- mented in an extended variant of the SDDiP algorithm to solve a basic single-echelon single-resource single-product uncapacitated (SULS) lot-sizing problem by Quezada, Gicquel, and Kedad-Sidhoum (2022). However, a direct implementation of this al- gorithm proved inefficient at solving the more realistic multi-echelon multi-resource multi-product lot-sizing problem investigated here. The main reason for this difficulty is that the sub-problems obtained through this partial decomposition are medium- size MILPs whose resolution by a mathematical solver requires a non-negligible time. We thus study in the present paper how the extended SDDiP algorithm proposed by Quezada, Gicquel, and Kedad-Sidhoum (2022) for a basic lot-sizing problem may be adapted to solve a more practical lot-sizing problem. This mainly involves reducing the computational effort needed to solve the above-mentioned sub-problems. Our ex- tensive computational results carried out on randomly generated instances show that the proposed approach can efficiently solve real-size instances of our remanufacturing planning problem. 1.1. Related literature A recent review of the literature on planning problems in remanufacturing may be found, e.g., in Kazemi, Modak, and Govindan (2019). Moreover, surveys on lot-sizing problems linked to remanufacturing were recently presented by Brahimi et al. (2017) and Suzanne, Absi, and Borodin (2020). This subsection focuses on the works dealing with dynamic lot-sizing for remanufacturing under uncertainty. We classify them according to three criteria: the production system, the uncertain 3 parameters, and the mathematical optimization approach used to handle the uncer- tainty in the production planning model. In terms of the production system, the simplest setting corresponds to the single- echelon single-product case. Such systems are investigated by Li et al. (2009), Kilic (2013), Kilic, Tunc, and Tarim (2018), Naeem et al. (2013) and Attila et al. (2021). Ex- tensions dealing with single-echelon multi-product systems are considered by Macedo et al. (2016), Hilger, Sahling, and Tempelmeier (2016) and He et al. (2022). To the best of our knowledge, there are four works dealing with multi-echelon multi-product systems: Wang and Huang (2013), Fang et al. (2017), Slama et al. (2022) and Frifita, Afsar, and Hnaien (2022). Wang and Huang (2013) consider a three-echelon system: acquisition of used products, disassembly of used products into parts and modules, reprocessing of the recovered parts and modules through repairing, remanufacturing or reusing. Fang et al. (2017) study a two-echelon system comprising a disassembly process followed by a hybrid manufacturing/remanufacturing process, whereas Frifita, Afsar, and Hnaien (2022) investigate a capacitated lot-sizing problem with disassembly and refurbishing operations. Finally, Slama et al. (2022) study a production sys-

tem that includes the disassembly and assembly operations of returned products, where lost sales are allowed with a penalty cost. We may also classify the papers according to the nature of the stochastic parame- ters. Wang and Huang (2013) and Fang et al. (2017) assume that the only stochastic parameter in the problem is the demand for remanufactured products. On the other hand, He et al. (2022) consider uncertain demand and components yield ratios, whereas Frifita, Afsar, and Hnaien (2022) only consider the case of the stochastic yield of the disassembly process. The case of stochastic demand and returns quantity is investigated by Li et al. (2009), Kilic (2013), Kilic, Tunc, and Tarim (2018), Naeem et al. (2013), Hilger, Sahling, and Tempelmeier (2016) and Attila et al. (2021). Macedo et al. (2016) extend these works by considering a stochastic demand, returns quantity and setup cost. Finally, the recent work of Slama et al. (2022) investigates the case of stochastic refurbishing lead times. Finally, various approaches may be used to handle the uncertainty in the mathemati- cal optimization model. Li et al. (2009) and Naeem et al. (2013) use stochastic dynamic programming approaches to minimize the total expected cost. These approaches rely on a set of discrete random variables, each one defined on a support comprising only three possible outcomes, to represent the uncertainties on the demand and returns quantity. Attila et al. (2021) and Frifita, Afsar, and Hnaien (2022) develop a robust optimization approach in which uncertainty is handled through uncertainty sets de- fined as budgeted polytopes. Two-stage stochastic integer programming approaches in which uncertainty is modeled through a set of sampled scenarios are investigated by Macedo et al. (2016), Hilger, Sahling, and Tempelmeier (2016), Wang and Huang (2013), He et al. (2022) and Slama et al. (2022). Multi-stage stochastic integer pro- gramming approaches are developed by Kilic (2013), Kilic, Tunc, and Tarim (2018) and Fang et al. (2017). The models developed by Kilic (2013) and Kilic, Tunc, and Tarim (2018) are based on the assumption that the decision process comprises several stages, each one corresponding to a planning period. At the first stage, i.e. at the be- ginning of the planning horizon, the periods of the planning horizon in which setups for manufacturing and/or remanufacturing may occur are determined. The following decision stages correspond to the beginning of each planning period. At the beginning of each period, the realization of the uncertain parameters up to that period is observed and, if a manufacturing/remanufacturing setup was scheduled for this period at the first decision stage, manufacturing/remanufacturing quantities are determined. Both 4 parameters is collected. More precisely, we consider a set of S = 1, . . . , decision stages. A decision stage may correspond to one or several planning periods: let T be the set of time periods belonging to stage . At the beginning of a decision stage, we assume that the information about the realization of the stochastic parameters is available up to (and including) that stage. Based on this information, we determine the setups and production quantities on each process for each planning period in T . 1 2 3 4 7 5 8 6 9 10 13 16 19 11 14 17 20 12 15 18 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 1 2 3 4 t = = 1 2 3 4 5 6 7 8 9 10 11 12 T = 12, = 4 V = 1, ..., 45 T 2 = 4, 5, 6 V6 = 6, 9 a6 = 5 t6 = 6 6 = 2 C(6) = 10, 13 V(6) = 6, 10, ..., 15, 22, ..., 33 Figure 2. Scenario tree structure Moreover, we assume that the evolution of the uncertain parameters can be repre- sented by a discrete scenario tree V. Each node n V belongs to a single period tn and a single stage n. Let Vt be the set of nodes related to period t. Each node n has a unique predecessor node denoted by an and represents the state of the system that can be distinguished by the information unfolded up to stage n. At any non-terminal node of the tree, there are one or several branches to indicate possible future outcomes of the random variables from the current node. Let C(n) be the set of children of node n and V(n) be the

sub-tree of V rooted in n. The probability associated with the state represented by node n is denoted by n. A scenario is a path in the tree from the root node to a leaf node and represents a possible outcome of the stochastic input param- eters over the whole planning horizon. Figure 2 illustrates these notations on a small scenario tree. Each node n V corresponds to a realization of the stochastic input parameters. Let rn be the quantity of collected used products, dn the demand for the remanufactured products and n i the proportion of recoverable parts i Ir obtained by disassembling one unit of returned product at node n V. As for the costs, we have the setup cost f n p for process p J , the unit inventory holding cost hn i for part i I, the unit lost-sales penalty cost ln, the unit cost qn i for discarding item i Ir 0. Finally, gn represents the cost of discarding the unrecoverable parts obtained while disassembly one unit of used products. As mentioned above, we assume that at each decision stage, 8 W n i 0 i Ir 0, n V (12) Xn p 0, Y n p 0, 1 p J , n V (13) The objective function (1) aims at minimizing the expected total cost, over all nodes of the scenario tree. The total cost at node n is the sum of the setup, inventory holding, lost sales and disposal costs. Constraints (2)-(5) represent the inventory balance con- straints for every node n V and every item i I. More specifically, Constraints (2) compute the physical inventory level of the used product i = 0 at each node n as the sum of the physical inventory level P San at the parent node of n and of the returns quantity rn minus the quantities, Xn 0 and W n 0 , of disassembled and discarded used products at node n. Constraints (3) state that the echelon inventory of each recover- able item i Ir is equal to the echelon inventory level ESan i at the parent node of n plus the number of items i recovered after the disassembly process, n i iXn 0 , minus the echelon demand for i to be actually satisfied, i(dn Ln) and the discarded quantity W n i . The same applies for the echelon inventory of the serviceable items i Is and the remanufactured product i = 2I +1 (Constraints (4)-(5)). Without loss of generality, we assume that the initial inventory, i.e. the entering inventory at the root node n = 1, is set to 0 of all items: see Con- straints (6)-(7). Constraints (8)-(9) ensure consistency be- tween the echelon inventory at consecutive levels of the bill-of-materials and guarantee that the physical inventory of each product remains non-negative. Constraints (10) link the production quantity variables to the setup variables. Finally, Constraints (11)-(13) provide the domain of the decision variables. For the sake of readability, we will use in what follows the vector Sn = (Sn 0 , Sn 1 , · · · , Sn 2I+1) to describe the leaving inventory at node n: Sn 0 = P Sn 0 repre- sents the physical inventory of item 0, Sn i = ESn i represents the echelon stock of item i I 0. Moreover, F n(Xn, Y n, Sn, W n, Ln) = pJ f n p Y n p + hn 0 Sn 0 + iIr hn i Sn i + iIs (hn i hn iI)Sn i + (hn 2I+1 iIr ihn i)Sn 2I+1 + lnLn + iIr 0 qn i W n i + gnXn 0 denotes the setup, inventory holding, lost sales and disposal cost at node n. Finally, Fn denotes the subset of constraints (8)-(13) related to node n. Note that small instances of Problem (1)-(13) can be directly solved by the standard branch-and-cut algorithm embedded in MILP solvers. Quezada et al. (2020) exploit the fact that, thanks to the use of echelon inventory variables, Problem (1)-(13) can be de- composed into a series of single-echelon lot-sizing sub-problems linked together by cou- pling constraints (8)-(9). They develop efficient customized branch-and-cut algorithms based on the use of problem-specific valid inequalities to strengthen the formulation of each single-echelon sub-problem and solve medium-size instances. Nonetheless, the size of the formulation grows exponentially fast with the number of nodes |V| in the scenario tree, leading to prohibitive computation times in practice for solving instances using large-size scenario trees. 2.4. Dynamic programming formulation In order to par- tially overcome this difficulty, we will investigate in Section 3 a new solution approach based on a partial nested decomposition of the original stochastic problem into a series

of smaller stochastic sub-problems. This solution approach uses as a starting point a dynamic programming reformulation of Problem (1)-(13) presented in this section. The proposed approach relies on a partial decomposition of the scenario tree V into a series of smaller sub-trees. This decomposition is obtained by first partitioning the 10

The problem studied here, i.e., stochastic production planning and lot-sizing for a three-echelon remanufacturing system, was previously investigated by Quezada, Gicquel, and Kedad-Sidhoum (2019a) and Quezada et al. (2020). Quezada et al. (2020) formulated the problem as a large-size mixed-integer linear program (MILP) and proposed a customized branch-and-cut algorithm based on new valid inequalities to solve it. Quezada, Gicquel, and Kedad-Sidhoum (2019a) later investigated the use of the Stochastic Dual Dynamic integer Programming (SDDiP) algorithm recently presented by Zou, Ahmed, and Sun (2019) to solve the problem. This algorithm relies on a full decomposition of the problem into a large number of small deterministic sub-problems. Although the above-mentioned solution approaches successfully provided near-optimal solutions for small to medium-size instances, some numerical difficulties were encountered to solve instances involving the large-size scenario trees which are often needed in real applications.

The present work thus discusses a new solution approach for this problem. This one uses a partial decomposition of the problem into a small number of medium-size stochastic sub-problems. Note that this partial decomposition was successfully implemented in an extended variant of the SDDiP algorithm to solve a basic single-echelon single-resource single-product uncapacitated (SULS) lot-sizing problem by Quezada, Gicquel, and Kedad-Sidhoum (2022). However, a direct implementation of this algorithm proved inefficient at solving the more realistic multi-echelon multi-resource multi-product lot-sizing problem investigated here. The main reason for this difficulty is that the sub-problems obtained through this partial decomposition are medium-size MILPs whose resolution by a mathematical solver requires a non-negligible time. We thus study in the present paper how the extended SDDiP algorithm proposed by Quezada, Gicquel, and Kedad-Sidhoum (2022) for a basic lot-sizing problem may be adapted to solve a more practical lot-sizing problem. This mainly involves reducing the computational effort needed to solve the above-mentioned sub-problems. Our extensive computational results carried out on randomly generated instances show that the proposed approach can efficiently solve real-size instances of our remanufacturing planning problem.

1.1. *Related literature*

A recent review of the literature on planning problems in remanufacturing may be found, e.g., in Kazemi, Modak, and Govindan (2019). Moreover, surveys on lot-sizing problems linked to remanufacturing were recently presented by Brahimi et al. (2017) and Suzanne, Absi, and Borodin (2020). This subsection focuses on the works dealing with dynamic lot-sizing for remanufacturing under uncertainty.

We classify them according to three criteria: the production system, the uncertain parameters, and the mathematical optimization approach used to handle the uncertainty in the production planning model.

In terms of the production system, the simplest setting corresponds to the single-echelon single-product case. Such systems are investigated by Li et al. (2009), Kilic (2013), Kilic, Tunc, and Tarim (2018), Naeem et al. (2013) and Attila et al. (2021). Extensions dealing with single-echelon multi-product systems are considered by Macedo

et al. (2016), Hilger, Sahling, and Tempelmeier (2016) and He et al. (2022). To the best of our knowledge, there are four works dealing with multi-echelon multi-product systems: Wang and Huang (2013), Fang et al. (2017), Slama et al. (2022) and Frifita, Afsar, and Hnaien (2022). Wang and Huang (2013) consider a three-echelon system: acquisition of used products, disassembly of used products into parts and modules, reprocessing of the recovered parts and modules through repairing, remanufacturing or reusing. Fang et al. (2017) study a two-echelon system comprising a disassembly process followed by a hybrid manufacturing/remanufacturing process, whereas Frifita, Afsar, and Hnaien (2022) investigate a capacitated lot-sizing problem with disassembly and refurbishing operations. Finally, Slama et al. (2022) study a production system that includes the disassembly and assembly operations of returned products, where lost sales are allowed with a penalty cost.

We may also classify the papers according to the nature of the stochastic parameters. Wang and Huang (2013) and Fang et al. (2017) assume that the only stochastic parameter in the problem is the demand for remanufactured products. On the other hand, He et al. (2022) consider uncertain demand and components yield ratios, whereas Frifita, Afsar, and Hnaien (2022) only consider the case of the stochastic yield of the disassembly process. The case of stochastic demand and returns quantity is investigated by Li et al. (2009), Kilic (2013), Kilic, Tunc, and Tarim (2018), Naeem et al. (2013), Hilger, Sahling, and Tempelmeier (2016) and Attila et al. (2021). Macedo et al. (2016) extend these works by considering a stochastic demand, returns quantity and setup cost. Finally, the recent work of Slama et al. (2022) investigates the case of stochastic refurbishing lead times.

Finally, various approaches may be used to handle the uncertainty in the mathematical optimization model. Li et al. (2009) and Naeem et al. (2013) use stochastic dynamic programming approaches to minimize the total expected cost. These approaches rely on a set of discrete random variables, each one defined on a support comprising only three possible outcomes, to represent the uncertainties on the demand and returns quantity. Attila et al. (2021) and Frifita, Afsar, and Hnaien (2022) develop a robust optimization approach in which uncertainty is handled through uncertainty sets defined as budgeted polytopes. Two-stage stochastic integer programming approaches in which uncertainty is modeled through a set of sampled scenarios are investigated by Macedo et al. (2016), Hilger, Sahling, and Tempelmeier (2016), Wang and Huang (2013), He et al. (2022) and Slama et al. (2022). Multi-stage stochastic integer programming approaches are developed by Kilic (2013), Kilic, Tunc, and Tarim (2018) and Fang et al. (2017). The models developed by Kilic (2013) and Kilic, Tunc, and Tarim (2018) are based on the assumption that the decision process comprises several stages, each one corresponding to a planning period. At the first stage, i.e. at the beginning of the planning horizon, the periods of the planning horizon in which setups for manufacturing and/or remanufacturing may occur are determined. The following decision stages correspond to the beginning of each planning period. At the beginning of each period, the realization of the uncertain parameters up to that period is observed and, if a manufacturing/remanufacturing setup was scheduled for this period at the first decision stage, manufacturing/remanufacturing quantities are determined. Both papers use random variables with a continuous probability distribution to represent the uncertainty on the demand and returns quantity. Fang et al. (2017) also consider a multi-stage decision process in which each decision stage corresponds to a planning period. However, their model differs from the ones of Kilic (2013) and Kilic, Tunc, and Tarim (2018) with respect to the decisions made at each stage. Fang et al. (2017) do not fix the setups for the whole planning horizon at the first stage but rather consider

that the decisions to be made at the beginning of each stage correspond to determining the disassembly/manufacturing/remanufacturing setups and production quantities for the corresponding planning period. Their model relies on a discrete scenario tree to represent the time evolution of the uncertain parameters.

In the present work, similar to what was done in Quezada, Gicquel, and Kedad-Sidhoum (2019a) and Quezada et al. (2020), we investigate production planning under uncertainty for a multi-echelon multi-product system involving disassembly, refurbishing and reassembly operations. We propose to handle this problem through a MSSiP approach in which the decisions to be made at each stage correspond to determining the setup and production quantities for the planning periods belonging to this stage. Our work is thus closely related to the one of Fang et al. (2017) who investigate a similar MSSiP approach for a multi-echelon multi-product remanufacturing system. However, Fang et al. (2017) only consider a stochastic demand, whereas we allow the demand, returns quantity, disassembly yield, and costs to be uncertain. Moreover, Fang et al. (2017) propose to solve the resulting MILP by a Lagrangian-based heuristic approach and solve instances with medium-size scenario trees involving 128 scenarios. In contrast, we investigate a decomposition method relying on a dynamic programming formulation of this stochastic problem and seek to solve instances with large-size scenario trees involving up to 3.2 million scenarios.

1.2. *Contributions*

In the present work, we study a multi-echelon multi-item multi-resource lot-sizing problem in order to plan production for a remanufacturing system. We simultaneously consider uncertainties on several input parameters: the demand for the remanufactured products, the returns quantity, the disassembly yield and the cost, the latter being induced by the uncertainties on the quality of the returns. Similar to Quezada, Gicquel, and Kedad-Sidhoum (2019a) and Quezada et al. (2020), we use a MSSiP approach to handle this stochastic combinatorial optimization problem. However, we seek to develop an algorithm capable of solving instances in which the evolution of the input parameters over time is represented by a very large-size scenario tree.

To this aim, we investigate how the extSDDiP algorithm introduced by Quezada, Gicquel, and Kedad-Sidhoum (2022) for solving the much simpler stochastic SULS problem may be adapted to solve the more realistic production planning problem considered here. Basically, the extSDDiP algorithm is an extension of the SDDiP algorithm introduced by Zou, Ahmed, and Sun (2019). It relies on a partial decomposition of the stochastic problem into a series of medium-size stochastic sub-problems linked together by expected cost-to-go functions. The algorithm provides near-optimal solutions of the initial problem by solving a sequence of stochastic sub-problems and iteratively building a piece-wise linear under-approximation of each expected cost-to-go function. This algorithm has a finite and optimal theoretical convergence and proves numerically efficient at solving the SULS problem. However, its direct implementation on the problem under study here leads to significant numerical difficulties. The main reason is that the stochastic sub-problems obtained when solving our remanufacturing planning problem with the extSDDiP algorithm are much larger than the ones obtained when solving the SULS problem so that the time needed to carry out one iteration of the extSDDiP algorithm becomes prohibitively long.

The main contribution of this paper thus consists in proposing an adaptation of the extSDDiP algorithm capable of solving a more realistic lot-sizing problem involving

multiple production echelons, multiple items and multiple resources. In a nutshell, the proposed algorithm relies on three main components which will be described in more detail in the remainder of the paper: a partial (rather than a full) decomposition of the problem into stochastic sub-problems, the use of continuous (rather than binary) state variables, and the generation of ε-optimal strengthened Benders' cuts to iteratively build piece-wise linear approximations of the expected cost-to-go functions.

Due to these last two components, the resulting algorithm will not be theoretically guaranteed to have a finite and optimal convergence. However, our numerical results show that, in practice, this approximate SDDiP algorithm leads to solutions of better quality and reduced computation time than the original SDDiP algorithm.

The remaining part of this paper is organized as follows. Section 2 describes the problem under study and introduces two mathematical formulations: an extensive MILP formulation and a nested dynamic programming formulation. The proposed approximate SDDiP algorithm is then presented in Section 3. Two further algorithmic enhancements, in particular the generation of ε-optimal strengthened Benders' cuts, are presented in Section 4. Computational experiments are reported in Section 5. Finally, conclusions and directions for further works are discussed in Section 7.

2. Problem description and modeling

2.1. *Remanufacturing system*

The production system comprises three production echelons (see Figure 1): disassembly of used products into recoverable parts, refurbishing of the recoverable parts into serviceable parts, and reassembly of serviceable parts into as-good-as-new remanufactured products. We aim at building a production-inventory plan for this system over a multi-period horizon consisting of a discrete set $\mathcal{T} = \{1, .., T\}$ of time periods.

The following assumptions are made on the remanufacturing system:

- A single type of used product, denoted as product 0, is returned in a limited quantity by customers in each time period $t \in \mathcal{T}$.
- The used products are disassembled into I types of parts. Let $\mathcal{I}_r = \{1, ..., I\}$ be the set of recoverable parts obtained by disassembly. The gozinto factor, i.e. the number of parts of type i embedded in a returned product, is denoted by ϖ_i.
- Each type of recoverable parts is refurbished on a dedicated resource to obtain serviceable parts. Let $I + 1$ denote the type of serviceable part obtained by refurbishing recoverable parts of type i. $\mathcal{I}_s = \{I + 1, ..., 2I\}$ is thus the set of serviceable parts.
- The serviceable parts are reassembled into a single type of remanufactured product, indexed by $2I + 1$, which is similar to the used product, i.e., which has the same bill-of-materials as the used product.
- The demand for the remanufactured product is dynamic, i.e., time-varying.
- Due to the usage state of the used products, some of the parts obtained during disassembly are not recoverable and have to be disposed of. The yield of the disassembly process, i.e., the proportion of parts that will be recoverable, is part-dependent, time-dependent, and smaller than 1.
- The yield of all other (refurbishing or reassembly) processes is constant and equal to 1.
- All production processes are uncapacitated.
- In case the demand for the remanufactured product cannot be satisfied on time

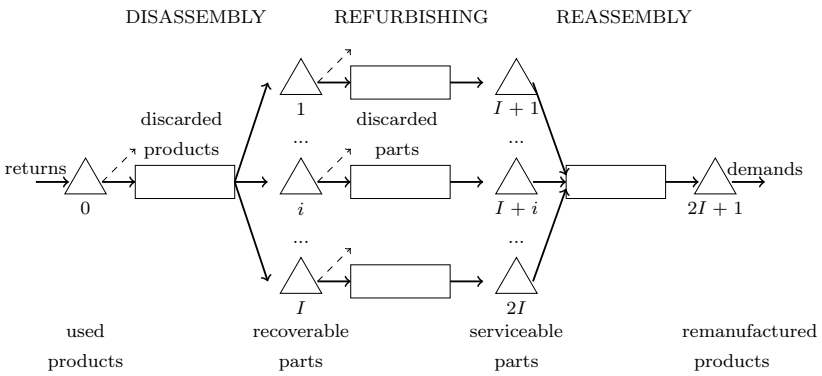

Figure 1. Illustration of studied remanufacturing system

(e.g., because we did not receive enough used products), the unmet demand is lost, and a high penalty cost is incurred to take into account the loss of customer goodwill.

- Returned products and recoverable parts can be discarded. This may be useful to avoid an unnecessary accumulation of inventory when more used products are returned than what is needed to satisfy the demand for remanufactured products and when the part-dependent disassembly yields are strongly unbalanced.

The remanufacturing system thus comprises a set $\mathcal{J} = \{0, ..., I+1\}$ of processes: $p = 0$ corresponds to the disassembly process, $p = i$ to the process refurbishing recoverable parts of type $i = 1, \cdots, I$ into serviceable parts of type $I + i$ and $p = I + 1$ to the reassembly process.

The goal is to find an optimal production plan, i.e., a production plan that minimizes the total production cost and satisfies all the practical limitations of the remanufacturing system. The total production cost comprises the fixed setup costs incurred each time a production takes place on a process, the inventory holding costs for all the items involved in the system, the lost-sales costs penalizing the unsatisfied demand, and the disposal costs for the discarded used products and parts.

Note that this production planning problem was previously investigated by Quezada et al. (2020). We thus refer the reader to this paper for more details.

2.2. *Uncertainty representation and management*

We consider the situation in which the input data needed to optimize the production plan for this system, in particular the demand, returns quantity, disassembly yield and costs, are subject to uncertainty. We propose to handle the resulting stochastic production planning problem using a MSSiP approach.

In this approach, production planning is seen as a multi-stage decision process in which some decisions have to be made at the beginning of the planning horizon, but others may be postponed until more information on the actual value of the input parameters is collected. More precisely, we consider a set of $\mathcal{S} = \{1, \ldots, \Sigma\}$ decision stages. A decision stage σ may correspond to one or several planning periods: let $\mathcal{T}^\sigma$ be the set of time periods belonging to stage σ. At the beginning of a decision stage, we assume that the information about the realization of the stochastic parameters is available up to (and including) that stage. Based on this information, we determine the setups and production quantities on each process for each planning period in $\mathcal{T}^\sigma$.

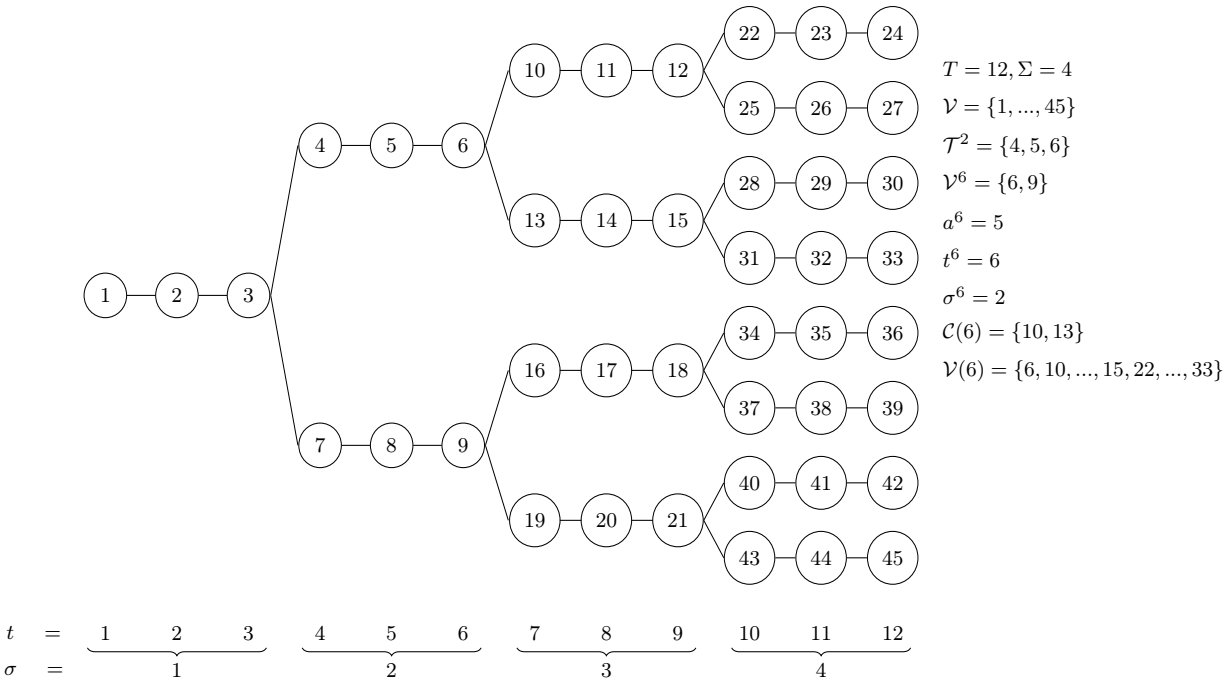

Figure 2. Scenario tree structure

Moreover, we assume that the evolution of the uncertain parameters can be represented by a discrete scenario tree $\mathcal{V}$. Each node $n \in \mathcal{V}$ belongs to a single period t^n and a single stage σ^n. Let $\mathcal{V}^t$ be the set of nodes related to period t. Each node n has a unique predecessor node denoted by a^n and represents the state of the system that can be distinguished by the information unfolded up to stage σ^n. At any non-terminal node of the tree, there are one or several branches to indicate possible future outcomes of the random variables from the current node. Let $\mathcal{C}(n)$ be the set of children of node n and $\mathcal{V}(n)$ be the sub-tree of $\mathcal{V}$ rooted in n. The probability associated with the state represented by node n is denoted by ρ^n. A scenario is a path in the tree from the root node to a leaf node and represents a possible outcome of the stochastic input parameters over the whole planning horizon. Figure 2 illustrates these notations on a small scenario tree.

Each node $n \in \mathcal{V}$ corresponds to a realization of the stochastic input parameters. Let r^n be the quantity of collected used products, d^n the demand for the remanufactured products and δ_i^n the proportion of recoverable parts $i \in \mathcal{I}_r$ obtained by disassembling one unit of returned product at node $n \in \mathcal{V}$. As for the costs, we have the setup cost f_p^n for process $p \in \mathcal{J}$, the unit inventory holding cost h_i^n for part $i \in \mathcal{I}$, the unit lost-sales penalty cost l^n, the unit cost q_i^n for discarding item $i \in \mathcal{I}_r \cup \{0\}$. Finally, g^n represents the cost of discarding the unrecoverable parts obtained while disassembly one unit of used products. As mentioned above, we assume that at each decision stage, the realization of the random parameters up to that stage is known before we have to make planning decisions for this stage. We thus assume that the values of r^n, d^n, δ_i^n, l^n, f_p^n, h_p^n, q_i^n and g^n are known for all nodes $n \in \cup_{t \in \mathcal{T}^\sigma} \mathcal{V}^t$ before we have to decide on the production plan for stage σ. We also assume that $l^n \gg g^n$ for all $n \in \mathcal{V}$.

2.3. *Mixed-integer linear programming formulation*

Based on the uncertainty representation described above, the stochastic problem can be formulated as a deterministic equivalent problem taking the form of a MILP. This formulation uses the concept of echelon stock. The echelon stock of a product in a multi-echelon production system corresponds to the total quantity of the product held in inventory, either as such or as a component within its successors in the bill-of-materials. In what follows, we will thus use the echelon stock to follow the total quantity of a product held in inventory for all products in $\mathcal{I}$ except product 0. Namely, due to the presence of stochastic and part-dependent disassembly yields, it is not possible to define an echelon stock for the used product $i = 0$. The reader is referred to Pochet and Wolsey (2006) for a general introduction on this concept and to Quezada et al. (2020) for a detailed description on how it can be used for the remanufacturing system under study.

We thus introduce the following decision variables for each node $n \in \mathcal{V}$:

- X_p^n: quantity of parts processed on process $p \in \mathcal{J}$,
- $Y_p^n \in \{0,1\}$: setup variable for process $p \in \mathcal{J}$,
- ES_i^n: echelon inventory of item $i \in \mathcal{I} \setminus \{0\}$,
- PS_0^n: physical inventory of used product $i = 0$,
- W_i^n: quantity of item $i \in \mathcal{I}_r \cup \{0\}$ discarded,
- L^n: lost sales of remanufactured products.

This leads to the following MILP formulation.

$$\min \sum_{n\in\mathcal{V}} \rho^n \Big(\sum_{p\in\mathcal{J}} f_p^n Y_p^n + h_0^n PS_0^n + \sum_{i\in\mathcal{I}_r} h_i^n ES_i^n + \sum_{i\in\mathcal{I}_s} (h_i^n - h_{i-I}^n) ES_i^n + (h_{2I+1}^n - \sum_{i\in\mathcal{I}_r} \varpi_i h_i^n) ES_{2I+1}^n + l^n L^n + \sum_{i\in\mathcal{I}_r\cup\{0\}} q_i^n W_i^n + g^n X_0^n \Big) \quad (1)$$

$$PS_0^n = PS_0^{a^n} + r^n - X_0^n - W_0^n \qquad \forall n \in \mathcal{V} \qquad (2)$$

$$ES_i^n = ES_i^{a^n} + \delta_i^n \varpi_i X_0^n - \varpi_i (d^n - L^n) - W_i^n \qquad \forall i \in \mathcal{I}_r, \forall n \in \mathcal{V} \qquad (3)$$

$$ES_i^n = ES_i^{a^n} + X_{i-P}^n - \varpi_i (d^n - L^n) \qquad \forall i \in \mathcal{I}_s, \forall n \in \mathcal{V} \qquad (4)$$

$$ES_{2I+1}^n = ES_{2I+1}^{a^n} + X_{P+1}^n - d^n + L^n \qquad \forall n \in \mathcal{V} \qquad (5)$$

$$PS_0^{a^1} = 0 \qquad (6)$$

$$ES_i^{a^1} = 0 \qquad \forall i \in \mathcal{I} \setminus \{0\} \qquad (7)$$

$$ES_i^n - ES_{I+i}^n \geq 0 \qquad \forall i \in \mathcal{I}_r, \forall n \in \mathcal{V} \qquad (8)$$

$$ES_i^n - \varpi_i ES_{2I+1}^n \geq 0 \qquad \forall i \in \mathcal{I}_s, \forall n \in \mathcal{V} \qquad (9)$$

$$X_p^n \leq M_p^n Y_p^n \qquad \forall p \in \mathcal{J}, \forall n \in \mathcal{V} \qquad (10)$$

$$ES_{2I+1}^n, L^n \geq 0 \qquad \forall n \in \mathcal{V} \qquad (11)$$

$$W_i^n \geq 0 \qquad \forall i \in \mathcal{I}_r \cup \{0\}, \forall n \in \mathcal{V} \qquad (12)$$

$$X_p^n \geq 0, Y_p^n \in \{0,1\} \qquad \forall p \in \mathcal{J}, \forall n \in \mathcal{V} \qquad (13)$$

The objective function (1) aims at minimizing the expected total cost, over all nodes of the scenario tree. The total cost at node n is the sum of the setup, inventory holding, lost sales and disposal costs. Constraints (2)-(5) represent the inventory balance con-

straints for every node $n \in \mathcal{V}$ and every item $i \in \mathcal{I}$. More specifically, Constraints (2) compute the physical inventory level of the used product $i = 0$ at each node n as the sum of the physical inventory level PS^{a^n} at the parent node of n and of the returns quantity r^n minus the quantities, X_0^n and W_0^n, of disassembled and discarded used products at node n. Constraints (3) state that the echelon inventory of each recoverable item $i \in \mathcal{I}_r$ is equal to the echelon inventory level $ES_i^{a^n}$ at the parent node of n plus the number of items i recovered after the disassembly process, $\delta_i^n \varpi_i X_0^n$, minus the echelon demand for i to be actually satisfied, $\varpi_i(d^n - L^n)$ and the discarded quantity W_i^n. The same applies for the echelon inventory of the serviceable items $i \in \mathcal{I}_s$ and the remanufactured product $i = 2I+1$ (Constraints (4)-(5)). Without loss of generality, we assume that the initial inventory, i.e. the entering inventory at the root node $n = 1$, is set to 0 of all items: see Constraints (6)-(7). Constraints (8)-(9) ensure consistency between the echelon inventory at consecutive levels of the bill-of-materials and guarantee that the physical inventory of each product remains non-negative. Constraints (10) link the production quantity variables to the setup variables. Finally, Constraints (11)-(13) provide the domain of the decision variables.

For the sake of readability, we will use in what follows the vector $S^n = (S_0^n, S_1^n, \cdots, S_{2I+1}^n)$ to describe the leaving inventory at node n: $S_0^n = PS_0^n$ represents the physical inventory of item 0, $S_i^n = ES_i^n$ represents the echelon stock of item $i \in \mathcal{I} \setminus \{0\}$. Moreover, $F^n(X^n, Y^n, S^n, W^n, L^n) = \sum_{p \in \mathcal{J}} f_p^n Y_p^n + h_0^n S_0^n + \sum_{i \in \mathcal{I}_r} h_i^n S_i^n + \sum_{i \in \mathcal{I}_s} (h_i^n - h_{i-I}^n) S_i^n + (h_{2I+1}^n - \sum_{i \in \mathcal{I}_r} \varpi_i h_i^n) S_{2I+1}^n + l^n L^n + \sum_{i \in \mathcal{I}_r \cup \{0\}} q_i^n W_i^n + g^n X_0^n$ denotes the setup, inventory holding, lost sales and disposal cost at node n. Finally, $\mathcal{F}^n$ denotes the subset of constraints (8)-(13) related to node n.

Note that small instances of Problem (1)-(13) can be directly solved by the standard branch-and-cut algorithm embedded in MILP solvers. Quezada et al. (2020) exploit the fact that, thanks to the use of echelon inventory variables, Problem (1)-(13) can be decomposed into a series of single-echelon lot-sizing sub-problems linked together by coupling constraints (8)-(9). They develop efficient customized branch-and-cut algorithms based on the use of problem-specific valid inequalities to strengthen the formulation of each single-echelon sub-problem and solve medium-size instances. Nonetheless, the size of the formulation grows exponentially fast with the number of nodes $|\mathcal{V}|$ in the scenario tree, leading to prohibitive computation times in practice for solving instances using large-size scenario trees.

2.4. *Dynamic programming formulation*

In order to partially overcome this difficulty, we will investigate in Section 3 a new solution approach based on a partial nested decomposition of the original stochastic problem into a series of smaller stochastic sub-problems. This solution approach uses as a starting point a dynamic programming reformulation of Problem (1)-(13) presented in this section.

The proposed approach relies on a partial decomposition of the scenario tree $\mathcal{V}$ into a series of smaller sub-trees. This decomposition is obtained by first partitioning the set of decision stages $\mathcal{S} = \{1, \ldots, \Sigma\}$ into a series of macro-stages $\mathcal{G} = \{1, \ldots, \Gamma\}$, where each macro-stage $\gamma \in \mathcal{G}$ contains a number of consecutive stages denoted by $\mathcal{S}(\gamma)$. We let $t(\gamma)$ (resp. $t'(\gamma)$) represent the first (resp. the last) time period belonging to macro-stage γ. For a given macro-stage γ, each node η belonging to the first time period in γ, i.e. each node $\eta \in \mathcal{V}^{t(\gamma)}$, is the root node of a sub-tree defined by the set of nodes $\mathcal{W}^\eta = \cup_{t=t(\gamma),\ldots,t'(\gamma)} \mathcal{V}^t \cap \mathcal{V}(\eta)$. $\mathcal{W}^\eta$ is thus the restriction of $\mathcal{V}(\eta)$ to the nodes

belonging to macro-stage γ. Let $\mathfrak{L}(\eta) = \mathcal{W}^\eta \cap \mathcal{V}^{t'(\gamma)}$ be the set of leaf nodes of sub-tree $\mathcal{W}^\eta$. Finally, we denote by $\mho = \cup_{\gamma \in \mathcal{G}} \mathcal{V}^{t(\gamma)}$ the set of sub-tree root nodes induced by $\mathcal{G}$.

These notations may be illustrated using the scenario tree depicted in Figure 2. Assume that the set of stages $\mathcal{S}$ is partitioned into $\Gamma = 2$ macro-stages with $\mathcal{S}(1) = \{1, 2\}$ and $\mathcal{S}(2) = \{3, 4\}$. The first time period of macro-stage $\gamma = 1$ is $t(1) = 1$, its last time period is $t'(1) = 6$. Similarly, we have $t(2) = 7$ and $t'(2) = 12$. In this case, the set of sub-tree root nodes is $\mho = \{1, 10, 13, 16, 19\}$. With this partition, node $\eta = 1$ is the root node of the subtree $\mathcal{W}^1 = \{1, 2, 3, 4, 5, 6, 7, 8, 9\}$ involving the set of leaf nodes $\mathfrak{L}(1) = \{6, 9\}$. Node $\eta = 10$ is the root node of sub-tree $\mathcal{W}^{10} = \{10, 11, 12, 22, 23, 24, 25, 26, 27\}$ involving the set of leaf nodes $\mathfrak{L}(10) = \{24, 27\}$. Sub-trees $\mathcal{W}^{13}$, $\mathcal{W}^{16}$ and $\mathcal{W}^{19}$ are defined in the same way as sub-tree $\mathcal{W}^{10}$.

The sub-problem $\mathscr{P}^\eta$ related to node $\eta \in \mho$ focuses on defining the production plan for the nodes belonging to sub-tree $\mathcal{W}^\eta$ based on the entering stock level S^{a^η} imposed by its parent node a^η in the scenario tree. $\mathscr{P}^\eta$ can be formulated as follows.

$$Q^\eta(S^{a^\eta}) \quad = \quad \min \sum_{n \in \mathcal{W}^\eta} \rho^n F^n(X^n, Y^n, S^n, W^n, L^n) \; + \sum_{\ell \in \mathfrak{L}(\eta)} \sum_{m \in \mathcal{C}(\ell)} Q^m(S^\ell) \quad (14)$$

$$S_0^n = S_0^{a^n} + r^n - X_0^n - W_0^n \qquad \forall n \in \mathcal{W}^\eta \qquad (15)$$

$$S_i^n = S_i^{a^n} + \delta_i^n \varpi_i X_0^n - \varpi_i d^n + \varpi_i L^n - W_i^n \qquad \forall i \in \mathcal{I}_r, \forall n \in \mathcal{W}^\eta \qquad (16)$$

$$S_i^n = S_i^{a^n} + X_{i-I}^n - \varpi_i d^n + \varpi_i L^n \qquad \forall i \in \mathcal{I}_s, \forall n \in \mathcal{W}^\eta \qquad (17)$$

$$S_{2I+1}^n = S_{2I+1}^{a^n} + X_{I+1}^n - d^n + L^n \qquad \forall n \in \mathcal{W}^\eta \qquad (18)$$

$$(X^n, Y^n, S^n, W^n, L^n) \in \mathcal{F}^n \qquad \forall n \in \mathcal{W}^\eta \qquad (19)$$

The objective function (14) comprises two terms: a term related to the expected production costs over sub-tree $\mathcal{W}^\eta$ and a term representing the future expected production costs incurred by the decisions made over sub-tree $\mathcal{W}^\eta$. More precisely, in (14), $Q^\eta(S^{a^\eta})$ denotes the optimal value of sub-problem $\mathscr{P}^\eta$ as a function of the entering stock level S^{a^η} and $Q^m(S^\ell)$ the optimal value of sub-problem $\mathscr{P}^m$ as a function of the entering stock level S^ℓ. The expected cost-to-go function at node $\ell \in \mathfrak{L}(\eta)$ is defined as the expected value of $Q^m(\cdot)$ over all the children of ℓ in the initial scenario tree $\mathcal{V}$, i.e. over all $m \in \mathcal{C}(\ell)$, which gives $\mathcal{Q}^\ell(\cdot) = \sum_{m \in \mathcal{C}(\ell)} Q^m(\cdot)$. The expected future costs of the decisions made in $\mathcal{W}^\eta$ are thus computed as the sum, over all nodes $\ell \in \mathfrak{L}(\eta)$, of $\mathcal{Q}^\ell(S^\ell)$. Note that for all leaf nodes, i.e. for all $\ell \in \mathcal{V}^T$, $\mathcal{Q}^\ell(\cdot) \equiv 0$.

We note that when $\mathcal{G} \equiv \mathcal{S}$, i.e. when each macro-stage corresponds to a single initial decision stage, each sub-tree $\mathcal{W}^\eta$ reduces to a set of nodes belonging to a single deterministic scenario involving T^{σ^η} periods. In this case, we obtain a decomposition similar to the one proposed by Zou, Ahmed, and Sun (2019) for general multi-stage stochastic integer programs and applied to the remanufacturing problem under study here by Quezada, Gicquel, and Kedad-Sidhoum (2019a).

3. Approximate Stochastic Dual integer Programming algorithm

The dynamic programming formulation (14)-(19) of the problem presented in Subsection 2.4 enables us to develop a solution approach based on the Stochastic Dual integer Programming (SDDiP) algorithm recently introduced by Zou, Ahmed, and Sun

(2019). Basically, this algorithm will solve Problem (1)-(13) by solving a sequence of sub-problems (14)-(19) in which each expected cost-to-go function $\mathcal{Q}^{\ell}(\cdot)$ is iteratively under-approximated by a piece-wise linear function.

The SDDiP algorithm is based on the stochastic dual dynamic programming (SDDP) algorithm, which has proved its usefulness at solving large-size multi-stage stochastic linear programs (Pereira and Pinto 1991). Note that for instance, the SDDP algorithm was recently used by Thevenin, Adulyasak, and Cordeau (2020) to solve a stochastic multi-echelon lot-sizing problem with component substitution. The authors assume that the setup decisions are determined in a pre-optimization step and use the SDDP algorithm to solve the resulting multi-stage stochastic linear program. The SDDiP algorithm is an extension of the SDDP algorithm developed by Zou, Ahmed, and Sun (2019) to solve multi-stage stochastic integer programs. Quezada, Gicquel, and Kedad-Sidhoum (2022) recently investigated how the SDDiP algorithm may be extended to efficiently solve large-size instances of the SULS problem. Preliminary attempts at solving the remanufacturing planning problem studied here with the SDDiP algorithm can be found in Quezada, Gicquel, and Kedad-Sidhoum (2019a) and Quezada, Gicquel, and Kedad-Sidhoum (2021b).

Both the SDDP and SDDiP algorithms rely on the key assumption that the scenario tree satisfies the stage-wise independence property Shapiro (2011). When a decision stage comprises several planning periods, this property can be defined as follows. Any two nodes m and m' belonging to the last period of stage $\sigma - 1$ (i.e. such that $t^m = t^{m'} = \max\{t, t \in \mathcal{T}^{\sigma-1}\}$) see the same potential evolutions of the uncertain parameters at stage σ. This means that the sets of nodes $\cup_{t\in\mathcal{T}^{\sigma}}\mathcal{V}^t \cap \mathcal{V}(m)$ and $\cup_{t\in\mathcal{T}^{\sigma}}\mathcal{V}^t \cap \mathcal{V}(m')$ are defined by identical data and conditional probabilities. Thanks to this property, the number of expected cost-to-go functions for which a piece-wise linear under-approximation must be built is significantly reduced. Namely, the expected cost-to-go functions relative to all nodes $m \in \mathcal{V}^{t'(\gamma)}$ are equal, i.e. we have $\mathcal{Q}^m(\cdot) \equiv \mathcal{Q}^{\gamma}(\cdot)$, for all $m \in \mathcal{V}^{t'(\gamma)}$. This means that we only have to build one expected cost-to-go function per macro-stage, rather than one expected cost-to-go function per node in $\mathcal{V}^{t'(\gamma)}$.

Moreover, we denote by $\mathcal{R}^{\gamma} = \{1, \ldots, R^{\gamma}\}$ the set of independent realizations of the stochastic process at macro-stage γ. Each realization $\mathcal{X}^{\gamma,\zeta}, \zeta \in \mathcal{R}^{\gamma}$, corresponds to a sub-tree describing a possible evolution of the uncertain parameters over periods $t(\gamma), \ldots, t'(\gamma)$. We denote by $\xi^{\gamma,\zeta}$ the root node of $\mathcal{X}^{\gamma,\zeta}$ and by $\mathfrak{L}(\gamma,\zeta)$ the set of its leaf nodes. Consequently, we can define a single sub-problem $\mathscr{P}^{\gamma}$ per macro-stage. Namely, for each sub-problem $\mathscr{P}^{\eta}, \eta \in \mho$, introduced in Subsection 2.4, we may identify the realisation $\mathcal{X}^{\gamma^{\eta},\zeta}$ corresponding to sub-tree $\mathcal{W}^{\eta}$ and describe $\mathscr{P}^{\eta}$ as $\mathscr{P}^{\gamma^{\eta}}(S^{a^{\eta}}, \mathcal{X}^{\gamma^{\eta},\zeta})$.

In what follows, we explain how the SDDiP algorithm proposed by Zou, Ahmed, and Sun (2019) may be extended to take into account a partial (rather than a full) decomposition of the problem into stochastic sub-problems. We also introduce two adaptations enabling us to reduce to some extent the computational effort related to the resolution of these sub-problems: the use of continuous (rather than binary) state variables and the generation of a single (rather than three) type of cuts.

3.1. *Sub-problem reformulation*

The proposed approximate SDDiP algorithm requires the introduction in the formulation of each problem $\mathscr{P}^{\gamma}(S^m, \mathcal{X}^{\gamma,\zeta})$, $\gamma \in \mathcal{G}$, of a set of auxiliary variables $\tilde{S}^{\xi^{\gamma,\zeta}}$. These variables can be seen as local copies of the variables S^m representing in $\mathscr{P}^{\gamma}(S^m, \mathcal{X}^{\gamma,\zeta})$ the leaving inventory at the parent node m. This results in the following reformulation

of sub-problem $\mathscr{P}^{\gamma}(S^m, \mathcal{X}^{\gamma,\zeta})$:

$$Q^{\gamma,\zeta}(S^m) \quad = \quad \min \sum_{n \in \mathcal{X}^{\gamma,\zeta}} \rho^n F^n(X^n, Y^n, S^n, W^n, L^n) \quad + \quad \sum_{\ell \in \mathfrak{L}(\gamma,\zeta)} \mathcal{Q}^{\gamma}(S^{\ell}) \quad (20)$$

$$S_0^{\xi^{\gamma,\zeta}} = \tilde{S}_i^{\xi^{\gamma,\zeta}} + r^{\xi^{\gamma,\zeta}} - X_0^{\xi^{\gamma,\zeta}} - Q_0^{\xi^{\gamma,\zeta}} \quad (21)$$

$$S_i^{\xi^{\gamma,\zeta}} = \tilde{S}_i^{\xi^{\gamma,\zeta}} + \delta_i^{\xi^{\gamma,\zeta}} \varpi_i X_0^{\xi^{\gamma,\zeta}} - \varpi_i d^{\xi^{\gamma,\zeta}} + \varpi_i L^{\xi^{\gamma,\zeta}} - Q_i^{\xi^{\gamma,\zeta}} \quad \forall i \in \mathcal{I}_r \quad (22)$$

$$S_i^{\xi^{\gamma,\zeta}} = \tilde{S}_i^{\xi^{\gamma,\zeta}} + X_{i-I}^{\xi^{\gamma,\zeta}} - \varpi_i d^{\xi^{\gamma,\zeta}} + \varpi_i L^{\xi^{\gamma,\zeta}} \quad \forall i \in \mathcal{I}_s \quad (23)$$

$$S_{2I+1}^{\xi^{\gamma,\zeta}} = \tilde{S}_{2I+1}^{\xi^{\gamma,\zeta}} + X_{I+1}^n - d^{\xi^{\gamma,\zeta}} + L^{\xi^{\gamma,\zeta}} \quad (24)$$

$$\tilde{S}_i^{\xi^{\gamma,\zeta}} = S_i^m \quad \forall i \in \mathcal{I} \quad (25)$$

$$S_0^n = S_0^{a^n} + r^n - X_0^n - Q_0^n \quad \forall n \in \mathcal{X}^{\gamma,\zeta} \setminus \{\xi^{\gamma,\zeta}\} \quad (26)$$

$$S_i^n = S_i^{a^n} + \delta_i^n \varpi_i X_0^n - \varpi_i d^n + \varpi_i L^n - Q_i^n \quad \forall i \in \mathcal{I}_r, \forall n \in \mathcal{X}^{\gamma,\zeta} \setminus \{\xi^{\gamma,\zeta}\} \quad (27)$$

$$S_i^n = S_i^{a^n} + X_{i-I}^n - \varpi_i d^n + \varpi_i L^n \quad \forall i \in \mathcal{I}_s, \forall n \in \mathcal{X}^{\gamma,\zeta} \setminus \{\xi^{\gamma,\zeta}\} \quad (28)$$

$$S_{2I+1}^n = S_{2I+1}^{a^n} + X_{I+1}^n - d^n + L^n \quad \forall n \in \mathcal{X}^{\gamma,\zeta} \setminus \{\xi^{\gamma,\zeta}\} \quad (29)$$

$$(X^n, Y^n, S^n, W^n, L^n) \in \mathcal{F}^n \quad \forall n \in \mathcal{X}^{\gamma,\zeta} \setminus \{\xi^{\gamma,\zeta}\} \quad (30)$$

In Formulation (20)-(30), Constraints (25) can be seen as copy constraints ensuring that each auxiliary variable is equal to the original variable it stands for. Moreover, note how the inventory balance equations at the root node $\xi^{\gamma,\zeta}$ of sub-tree $\mathcal{X}^{\gamma,\zeta}$ involve the auxiliary variables $\tilde{S}^{\xi^{\gamma,\zeta}}$ instead of the original variable S^m.

As explained above, the approximate SDDiP algorithm iteratively builds an under-approximation of each expected cost-to-go function $\mathcal{Q}^{\gamma}(\cdot)$ through a set of linear cuts.

Let $\tilde{\psi}_{\upsilon}^{\gamma}(\cdot)$ be the approximation of the expected cost-to-go function $\mathcal{Q}^{\gamma}(\cdot)$ available at iteration υ for macro-stage γ. We have:

$$\tilde{\psi}_{\upsilon}^{\gamma}(S^{\ell}) = \min\{\theta^{\gamma\ell} : \theta^{\gamma\ell} \geq \sum_{\zeta' \in \mathcal{R}^{\gamma+1}} (\tilde{\nu}_u^{\gamma+1,\zeta'} + \sum_{i \in \mathcal{I}} \tilde{\pi}_{u,i}^{\gamma+1,\zeta'} S_i^{\ell}) \quad \forall u \in \{1, ..., \upsilon - 1\}\} \quad (31)$$

where $\tilde{\nu}_u^{\gamma+1,\zeta'}$ and $\tilde{\pi}_{u,i}^{\gamma+1,\zeta'}$ are the coefficients of the cut generated at iteration $u < \upsilon$ by considering realization $\zeta' \in \mathcal{R}^{\gamma+1}$.

This leads to the following MILP sub-problem $\tilde{\mathscr{P}}_{\upsilon}^{\gamma}(S^m, \tilde{\psi}_{\upsilon}^{\gamma}, \mathcal{X}^{\gamma,\zeta})$:

$$\tilde{Q}_{\upsilon}^{\gamma,\zeta}(S^m) \quad = \quad \min \sum_{n \in \mathcal{X}^{\gamma,\zeta}} \rho^n F^n(X^n, Y^n, S^n, W^n, L^n) \quad + \quad \sum_{\ell \in \mathfrak{L}(\gamma,\zeta)} \theta^{\gamma\ell} \quad (32)$$

subject to

$$\theta^{\gamma\ell} \geq \sum_{\zeta' \in \mathcal{R}^{\gamma+1}} (\tilde{\nu}_u^{\gamma+1,\zeta'} + \sum_{i \in \mathcal{I}} \tilde{\pi}_{u,i}^{\gamma+1,\zeta'} S_i^{\ell}) \quad \forall u \in \{1, ..., \upsilon - 1\}, \quad \forall \ell \in \mathfrak{L}(\gamma, \zeta) \quad (33)$$

Constraints $(21) - (30)$

3.2. *Approximate SDDiP algorithm*

The approximate SDDiP algorithm is an iterative algorithm. Each iteration υ involves a sampling step, a forward step and a backward step.

In the sampling step, a sampling of K scenarios is carried out from the scenario tree. Let ω_υ^k be the set of nodes belonging to scenario k at iteration υ and $\Omega_\upsilon = \{\omega_\upsilon^1, \ldots, \omega_\upsilon^k, \ldots, \omega_\upsilon^K\}$ be the set of sampled scenarios. $\zeta_\upsilon^{k,\gamma}$ is the index of the realization in $\mathcal{R}^\gamma$ containing the values at macro-stage γ of the uncertain parameters in the scenario k sampled at iteration υ .

In the forward step, the algorithm proceeds stage-wise from macro-stage $\gamma = 1$ to Γ. For each sampled scenario ω_υ^k and each macro-stage γ, the problem $\tilde{\mathscr{P}}^\gamma(S_\upsilon^m, \tilde{\psi}_\upsilon^\gamma, \mathcal{X}^{\gamma,\zeta_\upsilon^{k,\gamma}})$ is solved. This problem uses the current approximation $\tilde{\psi}_\upsilon^\gamma$ of the expected cost-to-go function $\mathcal{Q}^\gamma(\cdot)$ and has an entering inventory S_υ^m computed at the node $m = \omega_\upsilon^k \cap \mathcal{V}^{t'(\gamma-1)}$ corresponding to the node in the sampled scenario ω_υ^k belonging to the last period of $\gamma - 1$. Before proceeding to the next macro-stage, we record S_υ^ℓ for the node $\ell = \omega_\upsilon^k \cap \mathfrak{L}(\gamma, \zeta_\upsilon^{k,\gamma})$. At the end of this step, the average and standard deviation of the total cost over all sampled scenarios are computed and these values are used to obtain a statistical upper-bound of the problem.

In the backward step, the algorithm proceeds stage-wise from macro-stage $\gamma = \Gamma$ to macro-stage 1. For each scenario $k = 1, \ldots, K$, each node $m \in \omega_\upsilon^k \cap \mathcal{V}^{t'(\gamma)}$ and each realization $\zeta \in \mathcal{R}^{\gamma+1}$, it solves a suitable relaxation of $\tilde{\mathscr{P}}^{\gamma+1}(S_\upsilon^m, \tilde{\psi}_{\upsilon+1}^{\gamma+1}, \mathcal{X}^{\gamma+1,\zeta})$. Note that this problem uses the updated approximation $\tilde{\psi}_{\upsilon+1}^{\gamma+1}$ of $\mathcal{Q}^{\gamma+1}(\cdot)$ which has just been computed and has an entering inventory S_υ^m corresponding to the value recorded for node m during the forward step. This relaxation is then used to generate new cuts and obtain an updated approximation $\tilde{\psi}_{\upsilon+1}^\gamma$ of $\mathcal{Q}^\gamma(\cdot)$. Finally, the sub-problem $\tilde{\mathscr{P}}^1(0, \tilde{\psi}_{\upsilon+1}^1, \mathcal{X}^{1,1})$ solved at macro-stage $\gamma = 1$ provides a lower bound for the overall problem. The algorithm stops when the upper and lower bounds are close enough, according to some convergence criterion.

The cuts generated during the backward step correspond to the strengthened Benders' cuts introduced by Zou, Ahmed, and Sun (2019). More precisely, we generate a cut for each macro-stage $\gamma = \Gamma - 1, \ldots, 1$ and each node $m \in \Omega_\upsilon \cap \mathcal{V}^{t'(\gamma)}$ as follows.

For each realization $\zeta' \in \mathcal{R}^{\gamma+1}$,

- We solve the linear relaxation of $\tilde{\mathscr{P}}_\upsilon^{\gamma+1}(S_\upsilon^m, \tilde{\psi}_{\upsilon+1}^{\gamma+1}, \mathcal{X}^{\gamma+1,\zeta'})$ to optimality. We get the dual value of each copy constraint (25) and use them to set the value of vector $\tilde{\pi}_\upsilon^{\gamma+1,\zeta'}$.
- We then solve the Lagrangian relaxation of $\tilde{\mathscr{P}}_\upsilon^{\gamma+1}(S_\upsilon^m, \tilde{\psi}_{\upsilon+1}^{\gamma+1}, \mathcal{X}^{\gamma+1,\zeta'})$ in which each copy constraint $\tilde{S}_i^{\xi^{\gamma,\zeta}} = S_{\upsilon,i}^m$ has been dualized with a Lagrangian multiplier set to $\tilde{\pi}_{\upsilon,i}^{\gamma+1,\zeta'}$. We record the optimal value of this Lagrangian relaxation and set $\tilde{\nu}_\upsilon^{\gamma+1,\zeta'}$ to this value.

This allows us to add to $\tilde{\psi}_\upsilon^\gamma$ a cut of type (34) for each node $m \in \Omega_\upsilon \cap \mathcal{V}^{t'(\gamma)}$ to obtain the improved approximation $\tilde{\psi}_{\upsilon+1}^\gamma$ of $\mathcal{Q}^{\gamma+1}(\cdot)$:

$$\theta^{\gamma,m} \geq \sum_{\zeta' \in \mathcal{R}^{\gamma+1}} \left(\tilde{\nu}_\upsilon^{\gamma+1,\zeta'} + \sum_{i \in \mathcal{I}} \tilde{\pi}_{\upsilon,i}^{\gamma+1,\zeta'} S_i^m \right) \tag{34}$$

As a synthesis, the main steps of the proposed approximate SDDiP algorithm are summarized in Algorithm 1.

Algorithm 1: Approximate SDDiP algorithm

Initialize $LB \leftarrow -\infty, UB \leftarrow +\infty, v \leftarrow 1$
while *no stopping criterion is satisfied* **do**
  Sampling step
  Randomly select K scenarios $\Omega_v = \{\omega_v^1, ..., \omega_v^K\}$
  Forward step
  for $k = 1, ..., K$ **do**
    for $\gamma = 1, ..., \Gamma$ **do**
      Solve $\tilde{\mathscr{P}}^\gamma(S_v^m, \tilde{\psi}_v^\gamma, \mathcal{X}^{\gamma,\zeta_v^{k,\gamma}})$ for $m = \omega_v^k \cap \mathcal{V}^{t'(\gamma-1)}$
      Record S_v^ℓ for $\ell = \omega_v^k \cap \mathfrak{L}(\gamma, \zeta_v^{k,\gamma})$
    end
    $cost^k \leftarrow \sum_{n \in \omega_v^k} F^n(X^n, Y^n, S^n, W^n, L^n)$
  end
  $\hat{\mu} \leftarrow \sum_{k=1}^K cost^k$ and $\hat{\sigma}^2 \leftarrow \frac{1}{K-1} \sum_{k=1}^K (cost^k - \hat{\mu})^2$
  $UB \leftarrow \hat{\mu} + z_{\alpha/2} \frac{\hat{\sigma}}{\sqrt{K}}$
  Backward step
  for $\gamma = \Gamma - 1, ..., 1$ **do**
    for $k = 1, ..., K$ **do**
      Let $m = \omega_v^k \cap \mathcal{V}^{t'(\gamma)}$
      for $\zeta \in \mathcal{R}^{\gamma+1}$ **do**
        Solve the linear relaxation of $\tilde{\mathscr{P}}^{\gamma+1}(S_v^m, \tilde{\psi}_{v+1}^{\gamma+1}, \mathcal{X}^{\gamma+1,\zeta})$ and collect the coefficients $\tilde{\pi}_v^{\gamma+1,\zeta'}$ of the strengthened Benders' cut
        Solve the Lagrangian relaxation of $\tilde{\mathscr{P}}^{\gamma+1}(S_v^m, \tilde{\psi}_{v+1}^{\gamma+1}, \mathcal{X}^{\gamma+1,\zeta})$ and collect the constant value $\tilde{\nu}_v^{\gamma+1,\zeta'}$ of the strengthened Benders' cut
      end
      Add the generated cut of type (34) to the current approximation of $\mathcal{Q}^\gamma$
    end
  end
  $LB \leftarrow \hat{Q}_{v+1}^1(0)$
  $v \leftarrow v + 1$
end

3.3. *Original vs approximate SDDiP algorithm*

The approximate SDDiP algorithm presented here differs from the original one with respect to three key features.

First, the original SDDiP algorithm is based on a full decomposition of the stochastic problem into small deterministic sub-problems. In this decomposition, each macro-stage γ corresponds to a single decision stage σ, there are thus Σ expected cost-to-go functions to approximate. In contrast, we propose to use a partial decomposition into medium-size stochastic sub-problems. This may positively impact the computational efficiency of the SDDiP algorithm. Namely, the number of expected cost-to-go functions to approximate is significantly reduced from the number of decision stages Σ to the number of macro-stages $\Gamma \ll \Sigma$. Furthermore, each sub-problem solved during the forward step of a given iteration covers a larger portion of the planning horizon and has a better visibility on the future input parameters. The obtained solution will thus tend to be less myopic and of better quality. All this may accelerate the global convergence of the algorithm. Yet, the size of the sub-problems to be solved at each iteration, and accordingly the computational effort needed to perform one iteration of the algorithm, will also increase. It is however possible to limit this increase to some extent: this will

be the purpose of the first algorithmic enhancement discussed in Subsection 4.1.

Second, Zou, Ahmed, and Sun (2019) show that the SDDiP algorithm has a finite and optimal convergence when the state variables, i.e. the variables linking the decision stages to one another, are restricted to be binary. However, in Problem (1)-(13), the state variables are the continuous inventory level variables S^n. In this case, Zou, Ahmed, and Sun (2019) suggested to use a binary approximation of the state variables but this would require the introduction of a large number of additional binary variables in the problem formulation. Our preliminary experiments showed that this leads to prohibitive computation times. In the proposed approximate SDDiP, we thus keep continuous state variables as done e.g. by Hjelmeland et al. (2019) and Quezada, Gicquel, and Kedad-Sidhoum (2019b). In this case, the finite convergence of the algorithm is not theoretically guaranteed anymore. However, in practice, this approximation may lead to better solutions thanks to the significant reduction of the computational effort required at each iteration.

Third, we only generate strengthened Benders' cuts to approximate the expected cost-to-go functions whereas the original SDDiP algorithm relies on three types of cuts: strengthened Benders' cuts, integer optimality cuts and Lagrangian cuts. Integer optimality cuts can only be generated when the state variables are binary. Moreover, the generation of Lagrangian cuts, even if possible in our case, would imply to solve a series of dual Lagrangian problems through a sub-gradient algorithm. The related computational effort leads to a significant increase in the computation time needed to carry out one iteration of the approximate SDDiP algorithm and negatively impacts its performance. In contrast, strengthened Benders' cuts can be generated with a limited computational effort. Moreover, we investigate in Subsection 4.2 a way of improving the quality of the solution obtained with Algorithm 1 by exploiting the existence of alternative MILP formulations for each sub-problem $\tilde{\mathscr{P}}^{\gamma+1}(S_v^m, \tilde{\psi}_{v+1}^{\gamma+1}, \mathcal{X}^{\gamma+1,\zeta})$ used in the backward step.

4. Algorithmic Enhancements

This section presents two ways of improving the numerical efficiency, both in terms of computation time and solution quality, of the approximate SDDiP algorithm discussed above.

4.1. *ε-optimal strengthened Benders' cuts*

The first enhancement aims at reducing the computational effort needed to generate the strengthened Benders' cuts in the backward step. This may be done by using a sub-optimal solution (instead of an optimal solution) of the Lagrangian relaxation of sub-problem $\tilde{\mathscr{P}}^{\gamma+1}(S_v^m, \tilde{\psi}_{v+1}^{\gamma+1}, \mathcal{X}^{\gamma+1,\zeta})$ to define the value of each constant $\tilde{\nu}_v^{\gamma+1,\zeta'}$ (see line 21 of Algorithm 1). Note that Rahmaniani et al. (2020) recently used a similar approach in the context of a Benders' decomposition algorithm.

We thus propose to use ε-optimal strengthened Benders' cuts, i.e. to generate strengthened Benders' cuts obtained by using a solution of the Lagrangian relaxation of each sub-problem $\tilde{\mathscr{P}}^{\gamma+1}(S_v^m, \tilde{\psi}_{v+1}^{\gamma+1}, \mathcal{X}^{\gamma+1,\zeta})$ which is at most ε units of the optimal value. This can be done as follows for a given iteration v, macro-stage γ and scenario k.

Let m be the node such that $m = \Omega_v \cap \mathcal{V}^{t'(\gamma)}$. For each realization $\zeta' \in \mathcal{R}^{\gamma+1}$,

- We solve the linear relaxation of $\tilde{\mathscr{P}}_v^{\gamma+1}(S_v^m, \tilde{\psi}_{v+1}^{\gamma+1}, \mathcal{X}^{\gamma+1,\zeta'})$ to optimality. We collect the dual value of each copy constraint (25) and use them to set the value of vector $\tilde{\pi}_v^{\gamma+1,\zeta'}$.
- We solve the Lagrangian relaxation of $\tilde{\mathscr{P}}_v^{\gamma+1}(S_v^m, \tilde{\psi}_{v+1}^{\gamma+1}, \mathcal{X}^{\gamma+1,\zeta'})$ in which each copy constraint $\tilde{S}_i^{\xi^{\gamma,\zeta}} = S_i^m$ has been dualized with a Lagrangian multiplier set to $\tilde{\pi}_{v,i}^{\gamma+1,\zeta'}$, until the value of the best known feasible solution is at most ε units from the best known lower bound. Let $\tilde{\nu}_v^{\gamma+1,\zeta'}(\varepsilon)$ be the value of this solution.

$\underline{\nu}_v^{\gamma+1,\zeta'}(\varepsilon) = \tilde{\nu}_v^{\gamma+1,\zeta'}(\varepsilon)(1-\varepsilon)$ is a lower bound of the optimal value of the Lagrangian relaxation of sub-problem $\tilde{\mathscr{P}}_v^{\gamma+1}(S_v^m, \tilde{\psi}_{v+1}^{\gamma+1}, \mathcal{X}^{\gamma+1,\zeta'})$, i.e., $\tilde{\nu}_v^{\gamma+1,\zeta'} \geq \underline{\nu}_v^{\gamma+1,\zeta'}(\varepsilon)$. The following cut is thus a valid linear under-approximation of $\mathcal{Q}^{\gamma}(S^m)$.

$$\theta^{\gamma m} \geq \sum_{\zeta' \in \mathcal{R}^{\gamma+1}} (\underline{\nu}_v^{\gamma+1,\zeta'}(\varepsilon) + \sum_{i \in \mathcal{I}} \tilde{\pi}_{v,i}^{\gamma+1,\zeta'} S_i^m) \tag{35}$$

4.2. *Generation of strengthened Benders' cuts using alternative MILP formulations*

We now investigate a way of improving the quality of the solution provided by the approximate SDDiP algorithm through the use of a wider set of strengthened Benders' cuts. The main idea is to exploit the fact that there exist alternative MILP formulations for the sub-problems $\tilde{\mathscr{P}}_v^{\gamma+1}(S_v^m, \tilde{\psi}_{v+1}^{\gamma+1}, \mathcal{X}^{\gamma+1,\zeta'})$.

Namely, $\tilde{\mathscr{P}}_v^{\gamma+1}(S_v^m, \tilde{\psi}_{v+1}^{\gamma+1}, \mathcal{X}^{\gamma+1,\zeta'})$ is a MILP whose initial formulation is given by Equations (32)-(33), (21)-(30). This formulation may be strengthened by two sets of valid inequalities: the (k, U) path inequalities presented by Quezada et al. (2020) or the stronger (ℓ, k, U) path inequalities recently introduced by Quezada, Gicquel, and Kedad-Sidhoum (2021a). The reader is referred to these two papers for more detail about these valid inequalities and the related separation algorithms.

Now, recall that in Algorithm 1, the coefficients $\tilde{\pi}_v^{\gamma+1,\zeta'}$ of the strengthened Benders' cut generated in the backward step are obtained by solving the linear relaxation of $\tilde{\mathscr{P}}_v^{\gamma}(S_v^m, \tilde{\psi}_v^{\gamma}, \mathcal{X}^{\gamma,\varsigma})$ and collecting the dual values of the copy constraints $\tilde{S}^{\xi^{\gamma,\varsigma}} = S^m$. A key observation here is that these dual values will vary according to the MILP formulation used for $\tilde{\mathscr{P}}_v^{\gamma}(S_v^m, \tilde{\psi}_v^{\gamma}, \mathcal{X}^{\gamma,\varsigma})$. Hence, for a given value of the entering stock S^m, by considering alternative MILP formulations for $\tilde{\mathscr{P}}_v^{\gamma}(S_v^m, \tilde{\psi}_v^{\gamma}, \mathcal{X}^{\gamma,\varsigma})$, it is possible to generate different strengthened Benders' cuts, each one corresponding to different values of coefficients $\tilde{\nu}_v^{\gamma,\zeta}$ and $\tilde{\pi}_v^{\gamma,\zeta}$.

We point out here that, in general, there does not exist a dominance relationship between these three different cuts. In other words, a cut generated using a stronger formulation of $\tilde{\mathscr{P}}_v^{\gamma}(S_v^m, \tilde{\psi}_v^{\gamma}, \mathcal{X}^{\gamma,\varsigma})$ does not necessarily lead to a better approximation of $\mathcal{Q}^{\gamma-1}(\cdot)$.

Similar to Quezada, Gicquel, and Kedad-Sidhoum (2022), we thus investigate an extension of Algorithm 1 in which strengthened Benders' cuts based on alternative MILP formulations for $\tilde{\mathscr{P}}_v^{\gamma}(S_v^m, \tilde{\psi}_v^{\gamma}, \mathcal{X}^{\gamma,\varsigma})$ are sequentially generated. The proposed strategy is based on three increasing levels of formulation improvement.

- At the initial level ($\lambda = 0$), the algorithm generates strengthened Benders' cuts based on the initial formulation of $\tilde{\mathscr{P}}_v^{\gamma}(S_v^m, \tilde{\psi}_v^{\gamma}, \mathcal{X}^{\gamma,\varsigma})$. It moves to the next level if the lower bound has not improved after a predefined number of consecutive

iterations.

- At the first level ($\lambda = 1$), the algorithm generates strengthened Benders' cuts based on the initial formulation of $\tilde{\mathscr{P}}_v^{\gamma}(S_v^m, \tilde{\psi}_v^{\gamma}, \mathcal{X}^{\gamma,\zeta})$ improved by (k, U) path inequalities. More precisely, at each iteration of the approximate SDDiP algorithm, we add (k, U) path inequalities to the formulation of each sub-problem using a single run of a cutting plane generation procedure proposed in Quezada et al. (2020) before solving its linear relaxation and collecting the dual values of the copy constraints. The algorithm moves to the next level when the lower bound has not improved after a predefined number of consecutive iterations.
- At the second level ($\lambda = 2$), the algorithm generates strengthened Benders' cuts based on the initial formulation strengthened by (ℓ, k, U) path inequalities. More precisely, at each iteration of the approximate SDDiP algorithm, we add (ℓ, k, U) path inequalities to the formulation of each sub-problem using a single run of a heuristic cutting plane generation procedure proposed by Quezada, Gicquel, and Kedad-Sidhoum (2021a) before solving the linear relaxation of the problem and collecting the dual values of the copy constraints. Finally, the algorithm stops if the lower bound has not improved after a predefined number of consecutive iterations.

5. Computational experiments

In this section, we focus on assessing the performance of the proposed approximate SDDiP algorithm by comparing it with the one of a stand-alone mathematical programming solver using the extensive formulation (1)-(13) and with the one of the original SDDiP algorithm introduced by Zou et al. Zou, Ahmed, and Sun (2019).

We first describe the scheme used to randomly generate instances of the stochastic problem. We then present in more detail the experimental setup used for our numerical experiments and discuss the corresponding computational results. Finally, we present the outcome of some rolling horizon simulations carried out on small-size instances. This enables us to draw some useful managerial insights regarding the impact of the scenario tree structure on the actual production cost.

5.1. *Instance generation*

We randomly generated instances as follows.

The number of parts was set to $I = 5$. For the bill-of-materials coefficients ϖ_i, we set $\varpi_0 = \varpi_{2I+1} = 1$ and randomly generated the value of $\varpi_i = \varpi_{i+I}$, $i = 1...I$, following a discrete uniform distribution over $[1; 6]$.

Regarding the scenario tree structure, we used only balanced trees with $\Sigma \in \{4, 6, 8, 12\}$ stages, a constant number $b \in \{1, 2, 3, 5\}$ of time periods per stage and a constant number $R \in \{3, 5, 10, 20\}$ of equi-probable realizations per stage. We considered 8 possible combinations for these parameters, leading to scenario trees involving between 1000 and 3.2 million scenarios. Costs were generated by using a production-holding cost ratio $g/h \in \{2, 4\}$, a setup-holding cost ratio $f/h \in \{200, 400\}$ and a returns-demand quantity ratio $r/d \in \{1, 3, 5\}$. For each considered scenario tree structure and each possible combination of g/h, f/h and r/d, five instances were randomly generated, resulting in a total of 480 instances.

More precisely, for each instance, we randomly generated the input data relative to

each node $n \in \mathcal{V}$ as follows.

- Demand d^n was uniformly distributed in the interval $[0, 100]$ and the returns quantity r^n was uniformly distributed in the interval $[0.8(r/d)\bar{d}, 1.2(r/d)\bar{d}]$, where $\bar{d} = \frac{1}{\mathcal{V}} \sum d^n$.
- The proportion of recoverable parts δ_i^n, $i \in \mathcal{I}_r$, was uniformly distributed in the interval $[0.4, 0.6]$.
- The holding cost h_0^n for the returned product $i = 0$ was fixed to 1. The holding cost h_i^n for each recoverable item $i \in \mathcal{I}_r$ was randomly generated following a discrete uniform distribution over interval $[2, 7]$. Similarly, the holding cost h_i^n for each serviceable item $i \in \mathcal{I}_r$ was randomly generated following a discrete uniform distribution over interval $[7, 12]$. Finally, in order to ensure non negative echelon costs, we set the value of the inventory holding cost for the remanufactured product, h_{2I+1}^n, to $\sum_{i=1}^{I} \varpi_i h_{I+i}^n + \epsilon$, where ϵ follows a discrete uniform distribution over interval $[80, 100]$.
- The production cost g^n was uniformly distributed in the interval $[0.8(g/h)\bar{h}, 1.2(g/h)\bar{h}]$, where $\bar{h} = \frac{1}{\mathcal{V}} \sum h_{2I+1}^n$.
- The setup cost f^n was uniformly distributed in the interval $[0.8(f/h)\bar{h}, 1.2(f/h)\bar{h}]$.
- Discarding costs for item $i \in \mathcal{I}_r \cup \{0\}$ were set to $q_i^n = 0.8\bar{h}_i^n$, where $\bar{h}_i^n = \frac{1}{|\mathcal{V}(n)|} \sum_{v \in \mathcal{V}(n)} h_i^v$.
- The unit penalty cost for lost sales, l^n, was fixed to 10000 per unit.
- The probability ρ^n is computed by assuming that all realizations at a given stage of the scenario tree are equiprobable. Therefore, $\rho^n = 1/R^{\sigma^n - 1}$.

5.2. *Experimental setup*

Each instance is first solved with the mathematical programming solver CPLEX 12.9 using the extensive MILP formulation (1)-(13). This solution method is denoted by CPX in what follows.

Each instance is then solved using the original SDDiP algorithm proposed by Zou, Ahmed, and Sun (2019). This algorithm is based on a full decomposition of the scenario tree in which each macro-stage comprises a single stage (i.e. $\Gamma = \Sigma$). Moreover, it uses the dynamic programming formulation (20)-(30) in which the continuous state variables are replaced by a set of binary variables thanks to a binary approximation. This one is carried out by computing an upper bound of the inventory level of item $i \in \mathcal{I}$ as $S_i^{max} = max_{\ell \in \mathcal{V}^T} \varpi_i d^{1,\ell}$. Each continuous variable $S_i^n, i \in \mathcal{I}, n \in \mathcal{V}$, is then approximated using a set of $B = \lceil log_2(\max_{i \in \mathcal{I}} S_i^{max}) \rceil$ binary variables.

Finally, each instance is solved using the proposed approximate SDDiP algorithm based on the dynamic programming formulation (20)-(30). We will refer to this algorithm as Algorithm appSDDiP. Several variants of this algorithm are considered in our numerical experiments. They differ from one another with respect to two aspects:

- The partial decomposition of the scenario tree. We use partitions of the set of decision stages $\mathcal{S}$ in which each macro-stage corresponds to a constant number $G \in \{1, 2\}$ of stages.
- The maximum level of formulation improvement considered when generating the strengthened Benders'cuts. Thus, appSDDiP-λ^{max} denotes a variant of the algorithm in which the sub-problem formulation improvement levels 0,...,λ^{max} are sequentially used following the strategy described at the end of Subsection 4.2.

Note that in Algorithm appSDDiP, all cuts are generated as ε-optimal strengthened Benders' cuts, with ε set to 1%.

Moreover, for the SDDiP and appSDDiP algorithms, the number of scenarios sampled at each iteration K is set to 1. We use the following stopping criteria: the algorithm stops when the lower bound LB does not improve after 30 iterations or when 1000 iterations have been carried out. Note that at this point, the upper bound UB is computed considering only $K = 1$ scenario and is not statistically representative. Thus, after the algorithm has stopped, we compute an updated statistical upper bound based on a larger number of scenarios as follows. We randomly sample $K' = 1000$ scenarios and compute a feasible solution for each of them using the final approximation of the expected cost-to-go functions to evaluate the objective function at each (macro)-stage. We then construct a 95% confidence interval and report the right endpoint of this interval as the statistical upper bound of the optimal value.

All the algorithms were implemented in C++ using the Concert Technology environment. The MILP and LP sub-problems embedded into the SDDiP and appSDDiP algorithms were solved using CPLEX 12.9. All computations have been carried out on the computing infrastructure of the Laboratoire d'Informatique de Paris VI (LIP6), which consists of a cluster of Intel Xeon Processors X5690. We set the cluster to use two 3.46GHz cores and 24GB RAM to solve each instance. We impose a time limit of 7200 seconds to method CPX to solve each instance. For the SDDiP and extSDDiP algorithms, we impose a time limit of 3600 seconds to compute a lower bound and 3600 seconds to compute the true or statistical upper bound.

5.3. *Results*

Tables 1-4 display the numerical results. Columns R and b describe the structure of the scenario tree when needed. The corresponding number of nodes in the scenario tree, $|\mathcal{V}|$, and the number of scenarios, $|\mathcal{V}^T|$, are then provided. Column G indicates the number of stages per macro-stage in the partial decomposition of the scenario tree and Column `Method` indicates the algorithm used to solve each instance (see notation at Subsection 5.2). Each line in the tables thus provides the average results of the indicated resolution method over the 60 instances corresponding to the given scenario tree structure but to various values of the r/d,f/h and g/h ratios. Column `Gap` displays the gap between the lower bound (LB) and the upper bound (UB) found by each method, i.e. $Gap = |UB - LB|/UB$. The average total computation time in seconds is reported in Column `Time(s)`, the average number of iterations in Column `#ite` and the total number of valid inequalities generated to improve the sub-problem formulation is provided in Column `#VI`.

Results from Table 1 first show that, when using the extensive formulation (1)-(13), method CPX outperforms the other methods for the smallest considered instances, i.e. the instances corresponding to $\Sigma = 4$, $R = 10$ and $b = 1$, providing an average gap of 0.24% within the allotted time limit. When the number of realizations per stage increases, i.e. for the instances corresponding to $\Sigma = 4$, $R = 20$ and $b = 1$, the relative performance of method CPX deteriorates but the average gap remains below 2%. However, when the number of stages, and consequently the size of the scenario tree, increases, the performance of method CPX strongly deteriorates. This can be seen from the results displayed in Tables 2-4: method CPX namely provides an average gap of 94% for the instances with $\Sigma = 8$, $R = 5$ and $b = 2$ and cannot find any feasible solution for the largest instances.

We also observe from the results displayed in Tables 2-4 that method SDDiP is able to provide feasible solutions for all the considered instances and that it consistently provides average gaps smaller than the ones obtained with method CPX for the medium to large-size instances. It thus clearly outperforms method CPX in terms of solution quality for these instances. However, the remaining gaps are still significant as they can be up to 55%.

Finally, these results show that the proposed appSDDiP algorithm significantly outperforms methods CPX and SDDiP. Namely, the average gap over all considered instances is significantly decreased from more than 50% with method CPX (resp. from 38.56% with method SDDiP) to 5.19% with method appSDDiP-2 and $G = 2$.

We now deepen our analysis and seek to independently assess the impact of each proposed adaptation of the initial SDDiP algorithm.

We first note that a large part of the gap reduction is obtained by using continuous (rather than binary) state variables. This can be seen by looking at the results obtained with the appSDDiP-0 algorithm for $G = 1$. We observe that the gap is reduced from 38.58% with method SDDiP to 8.70% with method appSDDiP-0. This improvement is mainly explained by the fact that the sub-problems expressed with continuous state variables are much easier to solve than the ones expressed with binary state variables. This allows the appSDDiP-0 algorithm to carry out more iterations than the SDDiP algorithm within the allotted time and to build better approximations of the expected cost-to-go functions.

We then study the impact of using a partial decomposition rather than a full decomposition of the scenario tree by comparing the results obtained with appSDDiP-0 for $G = 1$ and $G = 2$. We thus observe that the average gap is reduced from 8.70% with the appSDDiP-0 algorithm based a full decomposition ($G = 1$) to 5.67 % with the appSDDiP-0 algorithm based on a partial decomposition involving $G = 2$ stages per macro-stage. This shows the practical interest of using a partial decomposition of the scenario tree.

Finally, we can evaluate the impact of using valid inequalities to strengthen the linear relaxation of each sub-problem in order to generate additional strengthened Benders' cuts. Results in Tables 1-4 suggest that only a slight improvement in the performance of the extSDDiP algorithm is obtained by using improved MILP formulations for the sub-problems to generate strengthened Benders' cuts. The average gap is namely only reduced from from 5.67% with method appSDDiP-0 to 5.32% with method appSDDiP-

Table 1. Performance of each method at solving instances with $\Sigma = 4$ and $b = 1$

R	$\lvert\mathcal{V}\rvert$	$\lvert\mathcal{V}^T\rvert$	G	Method	Gap	Time (s)	# ite	# VI
10	1,110	1,000	1	SDDiP	23.55	4,306.65	50	0
				appSDDiP-0	7.46	1,358.58	222	0
			2	appSDDiP-0	2.00	205.15	87	0
				appSDDiP-1	1.20	1,696.08	173	566
				appSDDiP-2	1.18	1,956.50	192	580
			4	CPX	0.24	6,513.00	0	0
20	8,420	8,000	1	SDDiP	22.80	4,202.29	29	0
				appSDDiP-0	6.98	1,974.58	267	0
			2	appSDDiP-0	4.89	1,336.35	112	0
				appSDDiP-1	5.55	3,342.38	156	1,403
				appSDDiP-2	4.70	3,463.68	165	1,518
			4	CPX	1.57	7,201.81	0	0

Table 2. Performance of each method at solving instances with $\Sigma = 6$ and $b = 1$

R	$\lvert\mathcal{V}\rvert$	$\lvert\mathcal{V}^T\rvert$	G	Method	Gap	Time (s)	# ite	# VI
10	111,110	100,000	1	SDDiP	30.37	5,641.81	31	0
				appSDDiP-0	8.81	2,513.51	296	0
			2	appSDDiP-0	5.88	3,507.57	224	0
				appSDDiP-1	5.58	4,570.10	222	500
				appSDDiP-2	5.61	4,579.56	214	462
			4	CPX	68.27	7,217.53	0	0
20	3.36×10^6	3.2×10^6	1	SDDiP	35.67	5,608.54	19	0
				appSDDiP-0	9.83	3,664.12	395	0
			2	appSDDiP-0	8.45	4,599.68	131	0
				appSDDiP-1	7.66	4,894.02	109	84
				appSDDiP-2	7.59	4,814.77	111	84
			4	CPX	-	-	-	-

1 and to 5.19% method appSDDiP-2, for a partial decomposition based on $G = 2$.

6. Managerial insights

Before concluding, we would like to discuss some managerial insights relative to the practical implementation of an MSSiP approach such as the one studied in this paper.

6.1. *Data acquisition*

We first discuss the acquisition of the input data needed to build the scenario tree of the MSSiP. These data are mostly the same as the ones needed to implement other deterministic approaches as they consist in forecasts on the future demand, returns and costs. However, whereas deterministic approaches only need the expected value of each input parameter, the MSSiP approach requires probabilistic information about the distribution of each input parameter. In simple forecasting methods (e.g. exponential smoothing), this probabilistic information may be obtained by considering the forecasting error term. This one is usually modeled as a random variable following a centered normal distribution with a standard deviation quantifying the degree of accuracy of

Table 3. Performance of each method at solving instances with $\Sigma = 8$ and $R = 5$

b	$\lvert\mathcal{V}\rvert$	$\lvert\mathcal{V}^T\rvert$	G	Method	Gap	Time (s)	# ite	# VI
2	195,311	78,125	1	SDDiP	47.76	7,151.78	24	0
				appSDDiP-0	9.11	3,929.52	403	0
			2	appSDDiP-0	6.31	3,501.71	220	0
				appSDDiP-1	6.15	5,233.31	226	1,217
				appSDDiP-2	5.64	4,965.82	252	1,529
			4	CPX	93.48	7,236.10	0	0
5	488,279	78,125	1	SDDiP	41.74	7,222.84	14	0
				appSDDiP-0	6.68	3,815.32	378	0
			2	appSDDiP-0	4.11	5,884.19	140	0
				appSDDiP-1	4.42	6,667.21	123	1,612
				appSDDiP-2	4.73	6,335.19	146	2,657
			4	CPX	-	-	-	-

Table 4. Performance of each method at solving instances with $\Sigma = 12$ and $R = 3$

b	$\lvert\mathcal{V}\rvert$	$\lvert\mathcal{V}^T\rvert$	G	Method	Gap	Time (s)	# ite	# VI
1	265,719	177,147	1	SDDiP	51.04	7,207.72	38	0
				appSDDiP-0	11.00	3,313.90	356	0
			2	appSDDiP-0	7.69	2,567.41	402	0
				appSDDiP-1	6.34	3,807.25	424	514
				appSDDiP-2	6.29	3,595.10	505	685
			4	CPX	91.33	7,243.70	0	0
3	797,159	177,147	1	SDDiP	55.60	7,230.25	15	0
				appSDDiP-0	9.72	3,607.37	395	0
			2	appSDDiP-0	6.01	3,893.92	221	0
				appSDDiP-1	5.65	5,987.09	246	2,232
				appSDDiP-2	5.75	5,380.00	272	2,851
			4	CPX	-	-	-	-

the forecasts. More advanced forecasting methods (see e.g. ensemble learning methods) may directly provide probabilistic forecasts, i.e. forecasts assigning a probability to every possible outcome rather than providing a single value.

However, this probabilistic description often relies on continuous probability distributions. In order to build a scenario tree, we need to sample from this distribution. Brandimarte (2006) provides a numerical comparison between various sampling methods and recommends to use Latin hypercube sampling. Anyway, whatever the sampling method used to generate the scenario tree, there will clearly be a loss of information. The impact of this loss of information may be mitigated, at least partially, by increasing the number of children per node, i.e. the number of realizations per stage R. Namely, the more children per node in the scenario tree, the least information is lost during sampling and the better the quality of the obtained production plans. This however leads to a sharp increase in the size of the scenario tree, which shows the need to develop methods capable of handling large-size scenario trees.

6.2. *Quality of the stochastic solution*

The production planning model obtained with an MSSiP approach is more complex that the ones we would obtain with a deterministic approach (using the expected value of the random parameters as input data) or two-stage stochastic programming approach (considering the randomness of the input parameters but assuming that their actual value is revealed in one go once the first-stage decisions are made). Thus, the question arises about whether the quality of the stochastic solution obtained by an MSSiP approach is worth the additional modeling and computational effort. A first answer is provided by the results of the rolling-horizon simulation presented in Quezada, Gicquel, and Kedad-Sidhoum (2022). These results namely show that the average increase in the actual production cost observed when using the deterministic model (resp. the two-stage stochastic programming model) instead of the multi-stage stochastic model is 45% (resp. 19%). This clearly shows the potential benefit of using an MSSiP approach instead of simpler production planning models.

In what follows, we seek to gain some additional managerial insights on the impact of the scenario tree structure (defined by the number of stages and the number of realizations per stage) on the actual production cost observed when implementing the first-stage decisions recommended by the proposed MSSiP model.

This assessment is achieved by carrying out a rolling horizon simulation similar to the one used by Brandimarte (2006) and Quezada et al. (2020). This one consists in simulating, within a rolling horizon framework, the application of the first-stage decisions over a true scenario and in recording the corresponding cost.

In view of the related computational effort, we focus on instances involving small-size scenario trees and use a rolling horizon of $T^{sim} = 12$ time periods. The reference case corresponds to scenario trees with $\Sigma = 3$ decision stages, $b = 1$ period per stage and $R = 2$ realizations per stage. Let C_{ref} be the total actual cost, over the $T^{sim} = 12$ simulated periods, incurred by the application of the first-stage decisions over the true scenario. Note that this cost differs from the objective function value of the MSSiP model. We then seek to assess the impact of a change in the scenario tree structure on this actual cost. To this aim, we first fix Σ to 3 and increase R to 3, 4 or 5. Second, we set R to 2 and increase Σ to 4, 5 or 6. Let C_{mod} be the corresponding total actual cost. We finally compute the relative difference $RD = 100(C_{mod} - C_{red})/C_{red}$.

The instances were generated as in the previous subsection, apart from the disassembly yield δ which was generated from two uniform distributions: [0.1,0.5] (poor-quality returns) and [0.5,0.9] (good-quality returns). We randomly generated 400 true independent scenarios for each combination of the ratios $g/h \in \{2, 4\}$, $f/h \in \{200, 400\}$, $r/d \in \{1, 3\}$ and each level of returns quality, resulting in a total of 6400 true scenarios. Table 5 reports the average value and standard deviation of RD obtained with each modified scenario tree structure, each value of r/d and each level of returns quality.

The obtained results suggest the actual performance of the production plan provided by the MSSiP model might improve significantly when the number of realizations per stage R in the scenario tree increases. Namely, for $\Sigma = 3$, the actual production cost is decreased on average by 5.44% when R increases from 2 to 5. Moreover, we note that, for the instances with $r/d = 3$ and $\delta \in [0.1, 0.5]$, the cost decreases by more than 12% when R increases from 2 to 5. A detailed analysis shows that this cost reduction mainly comes from a decrease in the lost sales costs. This might be explained by the fact that with more realizations per stage in the scenario tree, we are able to more accurately anticipate the impact on the stochastic input parameters of our (in)ability at meeting the demand for the remanufactured products.

However, results from Table 5 also indicate that, in general, increasing the number of stages in the scenario tree leads to an increase of the actual cost of the production plan. This rather counter-intuitive result may be explained by the fact that the first-stage solutions provided by the MSSiP model are too conservative and tend to overproduce and/or overstock in order to gain protection against future realizations which will not actually occur in the true scenario.

Table 5. Impact on the actual production cost of the scenario tree structure

(Σ, R)		(3,3)		(3,4)		(3,5)		(4,2)		(5,2)		(6,2)	
r/d	δ	Ave.	Std.Dev.	Ave.	Std.Dev.	Ave.	Std.Dev.	Ave.	Std.Dev.	Ave.	Std.Dev.	Ave.	Std.Dev.
1	[0.1,0.5]	-1.33	3.45	-1.82	3.65	-2.05	3.42	0.99	4.24	2.52	4.50	4.74	5.39
	[0.6,0.9]	-3.70	6.66	-4.86	6.54	-5.64	6.62	0.10	7.28	2.55	8.45	3.19	7.84
3	[0.1,0.5]	-6.58	21.45	-10.61	17.93	-12.09	16.93	-4.58	20.45	-4.62	22.78	-7.35	19.49
	[0.6,0.9]	-1.02	8.94	-1.76	10.60	-1.98	11.43	5.55	12.40	10.28	15.70	14.86	18.30
Average		-3.15	12.41	-4.76	11.64	-5.44	11.62	0.52	13.18	2.68	15.56	3.86	16.22

7. Conclusion and perspectives

We studied production planning for a three-echelon remanufacturing system under uncertain input data. We investigated a scenario-tree based MSSiP approach and devoted ourselves to solving instances involving large-size scenario trees. To this aim, we proposed to adapt the extSDDiP algorithm investigated by Quezada, Gicquel, and Kedad-Sidhoum (2022) for the much simpler SULS problem and discussed several ways to overcome the numerical difficulties arising from the size of the sub-problems to be solved in the backward step of this algorithm. Our computational experiments carried out on large-size randomly generated instances suggested that the proposed algorithm is capable of obtaining near-optimal solutions in practicable computation times and outperforms both the mathematical programming solver CPLEX 12.9 using an extensive MILP formulation and the initial SDDiP algorithm proposed in Zou, Ahmed, and Sun (2019).

An interesting research direction could be to study whether the proposed algorithm may be useful at solving a larger class of lot-sizing problems involving e.g. capacitated resources, demand backlogging and/or non-zero setup times.

Acknowledgment

The authors would like to thank anonymous referees for their detailed reviews that helped to improve an initial version of this paper. This research was partially supported by Dicyt project 062217QV, Vicerrectoría de Investigación, Desarrollo e Innovación, Universidad de Santiago de Chile.

Data availability statement

The data that support the findings of this study are openly available in GitHub at https://github.com/FrancoQuezada/SDDiP_RLS_IJPR2022.git.

Disclosure statement

The authors report there are no competing interests to declare.

References

Attila, Öykü Naz, Agostinho Agra, Kerem Akartunalı, and Ashwin Arulselvan. 2021. "Robust formulations for economic lot-sizing problem with remanufacturing." *European Journal of Operational Research* 288 (2): 496–510.

Brahimi, Nadjib, Nabil Absi, Stéphane Dauzère-Pérès, and Atle Nordli. 2017. "Single-item dynamic lot-sizing problems: An updated survey." *European Journal of Operational Research* 263 (3): 838–863.

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

Fang, Chang, Xinbao Liu, Panos M Pardalos, Jianyu Long, Jun Pei, and Chao Zuo. 2017. "A stochastic production planning problem in hybrid manufacturing and remanufacturing systems with resource capacity planning." *Journal of Global Optimization* 68 (4): 851–878.

Frifita, Sana, Hasan Murat Afsar, and Faicel Hnaien. 2022. "A robust optimization approach for disassembly assembly routing problem under uncertain yields." *Expert Systems with Applications* 117304.

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

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

Hatcher, Gillian D, Winifred L Ijomah, and James FC Windmill. 2013. "Design for remanufacturing in China: a case study of electrical and electronic equipment." *Journal of Remanufacturing* 3 (1): 1–11.

He, Junkai, Feng Chu, Alexandre Dolgui, Feifeng Zheng, and Ming Liu. 2022. "Integrated stochastic disassembly line balancing and planning problem with machine specificity." *International Journal of Production Research* 60 (5): 1688–1708.

Hilger, Timo, Florian Sahling, and Horst Tempelmeier. 2016. "Capacitated dynamic production and remanufacturing planning under demand and return uncertainty." *OR Spectrum* 38 (4): 849–876.

Hjelmeland, Martin N., Jikai Zou, Arild Helseth, and Shabbir Ahmed. 2019. "Nonconvex Medium-Term Hydropower Scheduling by Stochastic Dual Dynamic Integer Programming." *IEEE Transactions on Sustainable Energy* 10 (1): 481–490.

Kazemi, Nima, Nikunja Mohan Modak, and Kannan Govindan. 2019. "A review of reverse logistics and closed loop supply chain management studies published in IJPR: a bibliometric and content analysis." *International Journal of Production Research* 57 (15-16): 4937–4960.

Kerr, Wendy, and Chris Ryan. 2001. "Eco-efficiency gains from remanufacturing: A case study of photocopier remanufacturing at Fuji Xerox Australia." *Journal of Cleaner Production* 9 (1): 75–81.

Kilic, Onur A. 2013. "A MIP-Based Heuristic for the Stochastic Economic Lot Sizing Problem with Remanufacturing." *IFAC Proceedings Volumes* 46 (9): 742–747.

Kilic, Onur A, Huseyin Tunc, and S Armagan Tarim. 2018. "Heuristic policies for the stochastic economic lot sizing problem with remanufacturing under service level constraints." *European Journal of Operational Research* 267 (3): 1102–1109.

Krystofik, Mark, Allen Luccitti, Kyle Parnell, and Michael Thurston. 2018. "Adaptive remanufacturing for multiple lifecycles: A case study in office furniture." *Resources, Conservation and Recycling* 135: 14–23.

Li, Congbo, Fei Liu, Huajun Cao, and Qiulian Wang. 2009. "A stochastic dynamic programming based model for uncertain production planning of re-manufacturing system." *International Journal of Production Research* 47 (13): 3657–3668.

Lund, Robert T. 1984. *Remanufacturing: the experience of the United States and implications for developing countries*. Vol. 31. World Bank.

Macedo, Pedro Belluco, Douglas Alem, Maristela Santos, Muris Lage Junior, and Alfredo Moreno. 2016. "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.

Naeem, Mohd, Dean J Dias, Rupak Tibrewal, Pei-Chann Chang, Manoj Kumar Tiwari, et al. 2013. "Production planning optimization for manufacturing and remanufacturing system in stochastic environment." *Journal of Intelligent Manufacturing* 24 (4): 717–728.

Pereira, Mario VF, and Leontina MVG Pinto. 1991. "Multi-stage stochastic optimization applied to energy planning." *Mathematical Programming* 52 (1-3): 359–375.

Pochet, Yves, and Laurence A Wolsey. 2006. *Production planning by mixed integer programming*. Vol. 149. Springer.

Quezada, Franco, Céline Gicquel, and Safia Kedad-Sidhoum. 2019a. "Stochastic Dual Dynamic integer Programming for a multi-echelon lot-sizing problem with remanufacturing and lost sales." In *2019 6th International Conference on Control, Decision and Information Technologies (CoDIT)*, 1254–1259.

Quezada, Franco, Céline Gicquel, and Safia Kedad-Sidhoum. 2019b. "A Stochastic Dual Dynamic Integer Programming for the Uncapacitated Lot-Sizing Problem with Uncertain Demand and Costs." In *Proceedings of the International Conference on Automated Planning and Scheduling*, Vol. 29, 353–361.

Quezada, Franco, Céline Gicquel, and Safia Kedad-Sidhoum. 2021a. "New Valid Inequalities for a Multi-echelon Multi-item Lot-Sizing Problem with Returns and Lost Sales." In *Computational Logistics*, edited by Martijn Mes, Eduardo Lalla-Ruiz, and Stefan Voß, Cham, 192–207. Springer International Publishing.

Quezada, Franco, Céline Gicquel, and Safia Kedad-Sidhoum. 2021b. "A partial nested decomposition approach for remanufacturing planning under uncertainty." In *IFIP International Conference on Advances in Production Management Systems*, 663–672. Springer.

Quezada, Franco, Céline Gicquel, and Safia Kedad-Sidhoum. 2022. "Combining polyhedral approaches and stochastic dual dynamic integer programming for solving the uncapacitated lot-sizing problem under uncertainty." *INFORMS Journal on Computing* 34 (2): 1024–1041.

Quezada, Franco, Céline Gicquel, Safia Kedad-Sidhoum, and Dong Quan Vu. 2020. "A multi-stage stochastic integer programming approach for a multi-echelon lot-sizing problem with returns and lost sales." *Computers & Operations Research* 116: 104865.

Rahmaniani, Ragheb, Shabbir Ahmed, Teodor Gabriel Crainic, Michel Gendreau, and Walter Rei. 2020. "The Benders dual decomposition method." *Operations Research* 68 (3): 878–895.

Rathore, Pragam, Srinivas Kota, and Amaresh Chakrabarti. 2011. "Sustainability through remanufacturing in India: a case study on mobile handsets." *Journal of Cleaner Production* 19 (15): 1709–1722.

Saavedra, Yovana MB, Ana PB Barquet, Henrique Rozenfeld, Fernando A Forcellini, and Aldo R Ometto. 2013. "Remanufacturing in Brazil: case studies on the automotive sector." *Journal of Cleaner Production* 53: 267–276.

Shapiro, Alexander. 2011. "Analysis of stochastic dual dynamic programming method." *European Journal of Operational Research* 209 (1): 63–72.

Slama, Ilhem, Oussama Ben-Ammar, Simon Thevenin, Alexandre Dolgui, and Faouzi Masmoudi. 2022. "Stochastic program for disassembly lot-sizing under uncertain component refurbishing lead times." *European Journal of Operational Research* .

Suzanne, Elodie, Nabil Absi, and Valeria Borodin. 2020. "Towards circular economy in production planning: Challenges and opportunities." *European Journal of Operational Research* 287 (1): 168–190.

Thevenin, Simon, Yossiri Adulyasak, and Jean-François Cordeau. 2020. "Stochastic Dual Dynamic Programming for Multi-Echelon Lot-sizing with Component Substitution." *Cahier du GERAD* 1–25.

Wang, Hsiao-Fan, and Yen-Shan Huang. 2013. "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.

Xiang, Wang, and Chen Ming. 2011. "Implementing extended producer responsibility: vehicle remanufacturing in China." *Journal of Cleaner Production* 19 (6-7): 680–686.

Zou, Jikai, Shabbir Ahmed, and Xu Andy Sun. 2019. "Stochastic dual dynamic integer programming." *Mathematical Programming* 175 (1): 461–502.

Appendix: Notation

Input parameters

I	Number of part types composing a returned/remanufactured product.
ϖ_i	Number of parts i embedded in a returned/remanufactured product.
$\mathcal{I}$	Set of all items involved in the remanufacturing production system.
$\mathcal{I}_r$	Set of recoverable parts provided by the disassembly process.
$\mathcal{I}_s$	Set of serviceable parts provided by the refurbishing processes.
$\mathcal{J}$	Set of production processes.
T	Number of time periods in the planning horizon.
$\mathcal{T}$	Set of time periods, i.e, $\mathcal{T} = \{1, \ldots, T\}$.
Σ	Number of decision stages in the stochastic process.
$\mathcal{S}$	Set of stages of the stochastic process, i.e., $\mathcal{S} = \{1, \ldots, \Sigma\}$.
$\mathcal{T}^\sigma$	Set of time periods belonging to stage $\sigma \in \mathcal{S}$.
$\mathcal{V}$	Set of nodes in a scenario tree.
$\mathcal{V}^t$	Set of nodes belonging to time period $t \in \mathcal{T}$.
$\mathcal{V}^\sigma$	Set of nodes belonging to stage $\sigma \in \mathcal{S}$.
$\mathcal{V}(n)$	Set of nodes belonging to the sub-tree of $\mathcal{V}$ rooted in node n.
ρ^n	Probability associated with the state represented by node n.
t^n	Time period $t \in \mathcal{T}$ of node $n \in \mathcal{V}$.
σ^n	Stage $\sigma \in \mathcal{S}$ of node $n \in \mathcal{V}$.
a^n	Predecessor of a node n in the scenario tree.
$\mathcal{C}(n)$	Set of immediate children of node $n \in \mathcal{V}$.
d^n	Customers' demand at node $n \in \mathcal{V}$.
r^n	Quantity of used products (returns) collected at node $n \in \mathcal{V}$.
δ_i^n	Proportion of recoverable parts $i \in \mathcal{I}_r$ obtained by disassembling one unit of returned product at node $n \in \mathcal{V}$.
f_p^n	Setup cost for process $p \in \mathcal{J}$ at node $n \in \mathcal{V}$.
h_i^n	Inventory holding cost for item $i \in \mathcal{I}$ at node $n \in \mathcal{V}$.
q_i^n	Unit cost for discarding item $i \in \mathcal{I}_r \cup \{0\}$ at node $n \in \mathcal{V}$.
g^n	Cost for discarding the unrecoverable parts obtained while disassembling one unit of returned product at node $n \in \mathcal{V}$.
l^n	Unit lost-sales penalty cost at node $n \in \mathcal{V}$.
$F^n(\cdot)$	Cost function at node $n \in \mathcal{V}$.
$\mathcal{F}^n$	Subset of Constraints (8)-(13) related to node n.

Decision Variables

ES_i^n	Echelon stock of item $i \in \mathcal{I} \setminus \{0\}$ at node $n \in \mathcal{V}$.
PS_0^n	Physical inventory level of used product $i = 0$ at node $n \in \mathcal{V}$.
S^n	Inventory level vector at node $n \in \mathcal{V}$: $S^n = (PS_0^n, ES_1^n, ES_2^n, \ldots, ES_{2I+1)}^n$.
L^n	Lost sales at node $n \in \mathcal{V}$.
W_i^n	Quantity of item $i \in \mathcal{I}_r \cup \{0\}$ discarded at node $n \in \mathcal{V}$.
X_p^n	Quantity to be produced by process $p \in \mathcal{J}$ at node $n \in \mathcal{V}$.
Y_p^n	Binary decision variable equal to 1 if there is setup on process $p \in \mathcal{J}$ at node $n \in \mathcal{V}$, to 0 otherwise.

Approximate SDDiP algorithm

Γ	Number of macro-stages.
$\mathcal{G}$	Set of macro-stages, i.e. partition of the set of decision stages $\mathcal{S}$.
$\mathcal{S}(\gamma)$	Set of consecutive stages belonging to macro-stage $\gamma \in \mathcal{G}$.
$t(\gamma)$	First time period belonging to macro-stage γ.
$t'(\gamma)$	Last time period belonging to macro-stage γ.
η	A node belonging to the first time period $t(\gamma)$ of macro-stage γ.
$\mathcal{W}^\eta$	Set of nodes of the sub-tree $\mathcal{V}(\eta)$ belonging to macro-stage γ, i.e., $\mathcal{W}^\eta = \cup_{t=t(\gamma),\ldots,t'(\gamma)} \mathcal{V}^t \cap \mathcal{V}(\eta)$.
$\mathfrak{L}(\eta)$	Set of leaf nodes of sub-tree $\mathcal{W}^\eta$, i.e, $\mathfrak{L}(\eta) = \mathcal{W}^\eta \cap \mathcal{V}^{t'(\gamma)}$.
$\mho$	Set of sub-tree root nodes induced by $\mathcal{G}$, i.e, $\mho = \cup_{\gamma \in \mathcal{G}} \mathcal{V}^{t(\gamma)}$.
$\mathscr{P}^\eta(\cdot)$	Sub-problem defined by the constraint and cost parameters at node $\eta \in \mho$.
$Q^\eta(\cdot)$	Optimal objective value of Problem $\mathscr{P}^\eta(\cdot)$.
$\mathcal{Q}^\ell(\cdot)$	Expected cost-to-go function at node $\ell \in \mathfrak{L}(\eta)$: $\mathcal{Q}^\ell(\cdot) = \sum_{m \in \mathcal{C}(\ell)} Q^m(\cdot)$.
$\mathcal{Q}^\gamma(\cdot)$	Expected cost-to-go function at macro-stage $\gamma \in \mathcal{G}$.
F^γ	Number independent realizations at macro-stage γ.
$\mathcal{R}^\gamma$	Set of independent realizations at macro-stage γ.
$\mathcal{X}^{\gamma,\zeta}$	Sub-tree describing realization $\zeta \in \mathcal{R}^\gamma$.
$\xi^{\gamma,\zeta}$	Root node of sub-tree $\mathcal{X}^{\gamma,\zeta}$.
$\mathfrak{L}(\gamma,\zeta)$	Leaf nodes of sub-tree $\mathcal{X}^{\gamma,\zeta}$.
$\mathscr{P}^\gamma(\cdot,\cdot)$	Sub-problem defined at macro-stage γ.
$\tilde{S}_i^{\xi^{\gamma,\zeta}}$	Auxiliary decision variable corresponding to a local copy in Problem $\mathscr{P}^\gamma(\cdot,\cdot)$ of the inventory level variable S_i^m.
$\tilde{\psi}_\upsilon^\gamma(\cdot)$	Approximation of the expected cost-to-go function $\mathcal{Q}^\gamma(\cdot)$ available at iteration υ of the approximate SDDiP algorithm for macro-stage γ.
$\theta^{\gamma,\ell}$	Decision variable taking the minimum value among a set of available linear supporting hyper-planes.
$\tilde{\pi}_{\upsilon,i}^{\gamma+1,\zeta'}$	Coefficient of inventory variable S_i^ℓ in the cut generated at iteration υ of the approximate SDDiP algorithm considering realization $\zeta' \in \mathcal{R}^{\gamma+1}$.
$\tilde{\nu}_\upsilon^{\gamma+1,\zeta'}$	Constant coefficient in the cut generated at iteration υ of the approximate SDDiP algorithm considering realization $\zeta' \in \mathcal{R}^{\gamma+1}$.
$\underline{\nu}_\upsilon^{\gamma+1,\zeta'}(\varepsilon)$	Constant coefficient in the ε-optimal strengthened Benders' cut generated at iteration υ of the approximate SDDiP algorithm considering realization $\zeta' \in \mathcal{R}^{\gamma+1}$.
$\tilde{\mathscr{P}}_\upsilon^\gamma(\cdot,\cdot,\cdot)$	Approximate MILP sub-problem related to macro-stage γ solved at iteration υ of the approximate SDDiP algorithm.
K	Number of sampled scenarios in the forward step.
Ω_υ	Set of sampled scenarios in the forward step of the SDDiP algorithm at iteration υ.
ω_υ^w	A sampled scenario in Ω_υ at iteration υ.

List of figures

Figure 1 Caption: Illustration of studied remanufacturing system.
Figure 1 Alt Text: Diagram describing the flow of returned products inside the remanufacturing production system.
Figure 1 Long Description: Diagram describing the flow of returned products inside the remanufacturing production system; each stock level is represented by a triangle and each process is represented by a rectangle. The flow of returned products through the system is represented by arrows that connect stock levels and processes.

Figure 2 Caption: Scenario tree structure.
Figure 2 Alt Text: Set of circles connected by straight lines.
Figure 2 Long Description: Set of circles connected by straight lines. Each circle represents a possible realization of the stochastic parameters at each time period and each line connects the possibles realizations from one period to the next. A Notation example on a small scenario tree is included in the figure.