**Les Cahiers du GERAD** ISSN: 0711–2440

# Stochastic dual dynamic programming for multi-echelon lot-sizing with component substitution

S. Thevenin,
Y. Adulyasak, J.-F. Cordeau

G–2020–64

November 2020

---



---

La publication de ces rapports de recherche est rendue possible grâce au soutien de HEC Montréal, Polytechnique Montréal, Université McGill, Université du Québec à Montréal, ainsi que du Fonds de recherche du Québec – Nature et technologies.

Dépôt légal – Bibliothèque et Archives nationales du Québec, 2020
– Bibliothèque et Archives Canada, 2020

The publication of these research reports is made possible thanks to the support of HEC Montréal, Polytechnique Montréal, McGill University, Université du Québec à Montréal, as well as the Fonds de recherche du Québec – Nature et technologies.

Legal deposit – Bibliothèque et Archives nationales du Québec, 2020
– Library and Archives Canada, 2020

---

**GERAD** HEC Montréal
3000, chemin de la Côte-Sainte-Catherine
Montréal (Québec) Canada H3T 2A7

**Tél. : 514 340-6053**
Téléc. : 514 340-5665
info@gerad.ca
www.gerad.ca

# Stochastic dual dynamic programming for multi-echelon lot-sizing with component substitution

**Simon Thevenin** [a,b]

**Yossiri Adulyasak** [b,c]

**Jean-François Cordeau** [b,c]

[a] *GERAD, Montréal (Québec), Canada, H3T 2A7*

[b] *IMT Atlantique, LS2N–CNRS, 44307 Nantes, France*

[c] *Department of Logistics and Operations Management, HEC Montréal, Montréal (Québec), Canada, H3T 2A7*

simon.thevenin@imt-atlantique.fr
yossiri.adulyasak@hec.ca
jean-francois.cordeau@hec.ca

**November 2020**
**Les Cahiers du GERAD**
**G–2020–64**

**Abstract:** This work investigates lot-sizing with component substitution under demand uncertainty. The integration of component substitution with lot-sizing in an uncertain demand context is important because the consolidation of the demand for components naturally allows risk pooling and reduces operating costs. The considered problem is not only relevant in a production context, but also in the context of distribution planning. We propose a stochastic programming formulation for the static-dynamic type of uncertainty, where the setup decisions are frozen, but the production and consumption quantities are decided dynamically. To tackle the scalability issues commonly encountered in multi-stage stochastic optimization, this paper investigates the use of stochastic dual dynamic programming (SDDP). In addition, we consider various improvements of SDDP, including the use of strong cuts, the fast generation of cuts by solving the linear relaxation of the problem, and retaining the average demand scenarios. Finally, we propose two heuristics, namely, a hybrid of progressive hedging with SDDP, and a heuristic version of SDDP. Computational experiments conducted on well-known instances from the literature show that the heuristic version of SDDP outperforms other methods. Indeed, the proposed method can plan with up to ten decision stages and 20 scenarios per stage, which results in $20^{10}$ possible scenarios in total. Finally, as the heuristic version of SDDP can re-plan to account for new information in less than a second, it is convenient in a dynamic context.

**Keywords:** Lot-sizing, stochastic dual dynamic programming, stochastic optimization, stochastic demand

**Acknowledgments:** This research was supported by the Institut de Valorisation des Données (IVADO) and by MITACS. The authors would also like to thank the CCIPL computing center and the region Pays de la Loire for the generous computing resource allocation.

# 1 Introduction

This paper investigates the multi-echelon lot-sizing problem with component substitution under uncertain demand. Unlike what is commonly assumed in production planning, the bills of materials of various manufacturing products found in practice are usually not fixed. For instance, in the metal industry, end-items are often cut from any large enough piece of metal. In fact, component substitution is common in various industries such as computers (Begnaud et al. 2009) and semi-conductors (Rao et al. 2004). Lot-sizing with substitution also occurs in distribution planning, when a retailer can be supplied by different distribution centers with different transportation costs.

The integration of component selection with lot-sizing in an uncertain demand context is important since the consolidation (centralization) of the demand for components reduces the fixed setup costs and allows risk pooling. In such circumstances, the decision maker can benefit from the flexibility through an adaptive decision process. That is, the manufacturer decides the amount of end-item to produce and the consumed components dynamically, after observing the demand in the previous period. Consequently, we consider the case of stochastic demand with static-dynamic types of uncertainty.

In the static-dynamic type of uncertainty (Bookbinder and Tan 1988), the setups are decided in the initial period (period 0) for the entire horizon, whereas the lot sizes are chosen at each period after having observed the demand. This decision process leads to a multi-stage stochastic optimization problem, and such problems are commonly modeled with a scenario tree. However, scenario trees have two major drawbacks: (1) the size of the tree grows exponentially with the number of decision stages (leading to $N^T$ scenarios where $N$ is the number of scenarios in each stage, and $T$ is the number of decision stages), and this is impractical for a long planning horizon; (2) the solution computed based on a sampled scenario tree gives only the first stage decisions, and the problem must be solved again from scratch when new information (not in the considered sample) is available.

To overcome these difficulties, the present paper investigates the use of the stochastic dual dynamic programming (SDDP) algorithm framework. This framework was proposed by Pereira and Pinto (1991) to solve multi-stage stochastic optimization problems. SDDP decomposes the problem per decision stage, and it iterates successive forward and backward passes. In the forward pass, the decisions in each stage $t$ are computed with the current policy based on the state of the system, on the cost of stage $t$, and on the approximated cost-to-go (from stage $t+1$ to the end of the horizon). The backward pass improves the cost-to-go approximations with information on the dual sub-problems. Note that SDDP does not suffer from the drawbacks of the scenario tree approach for the following reasons. First, the stage-wise decomposition in SDDP prevents the exponential growth of the number of scenarios (as a result, SDDP considers only $N$ scenarios per stage), provided that the probability distributions of the different decision stages are independent. Second, the cost-to-go approximations remain valid for any realization of the stochastic parameters. Therefore, when new information unfolds, SDDP updates the quantities in a fraction of a second, since it only requires to solve the sub-problem of the corresponding stage. However, building accurate approximations of the future costs in each stage requires a lot of computation, and it is a very time consuming process.

The contribution of this work lies in the study of SDDP for the multi-echelon lot-sizing problem with component substitution under static-dynamic uncertain demands. To the best of our knowledge, we are the first to consider this problem. Several enhancements of SDDP are studied, namely, the use of strong cuts (Magnanti and Wong 1981) with the implementation of Papadakos (2008), the incorporation of a lower bound computed with the average demand scenario, the multi-cut version of SDDP, and scenario sampling with Randomized Quasi-Monte Carlo (Cranley and Patterson 1976). As the setup variables are binary, they considerably slow down the convergence of SDDP. Consequently, we propose two strategies to compute and fix the setups, namely, a hybrid of progressive hedging with SDDP, and a heuristic version of SDDP. To study the performance of these methods, some experiments are conducted with well-known instances from Tempelmeier and Derstroff (1996). The results show

that the heuristic SDDP outperforms the hybrid of progressive hedging (PH) with SDDP and the two-stage heuristic proposed in Thevenin et al. (2020).

The rest of this paper is organized as follows. Section 2 reviews the literature on component substitution and stochastic optimization in lot-sizing. Section 3 describes the multi-echelon capacitated lot-sizing problem under static-dynamic uncertain demands. Section 4 presents the SDDP algorithms, and the methods to compute the setups are presented in Section 5. Finally, Section 6 reports the experimental results, followed by a conclusion in Section 7.

# 2 Literature review

Since the seminal paper of Harris (1990) (reprint of the 1913 article), the lot-sizing problem has received considerable attention from the operations research community. Readers interested in the latest development on this topic are referred to recent surveys (e.g., Karimi et al. 2003, Jans and Degraeve 2008, Buschkühland et al. 2010, Brahimi et al. 2017). The rest of this section reviews the specific characteristic considered in this work, namely, component substitution and stochastic demand.

## 2.1 Literature review on lot-sizing with component substitution

In various manufacturing contexts, producers have the flexibility to substitute a given component by a component of better quality or with more functionalities. Despite the higher cost of these alternative components, the substitution is relevant since it reduces the ordering costs. Similarly, in a transportation context, a local distribution center can be substituted by a farther one at the expense of additional transportation costs. Lot-sizing with component substitution was introduced by Balakrishnan and Geunes (2000). The authors propose a dynamic programming approach, and they show that component substitution can lead to significant cost savings, especially in a dynamic demand context.

Component selection is usually considered in a single echelon bill of material (BOM). Geunes (2003) proposes a facility location reformulation to solve the problem with the dual based procedure proposed by Erlenkotter (1978) as a solution method. Yaman (2009) considers the specific case with only two items and one-way substitution. The author provides a polyhedral analysis and gives some valid inequalities. Later, Lang and Domschke (2010) introduced the case where multiple components are required to substitute a component. The authors propose a facility location reformulation, as well as some valid inequalities. A different stream of research (e.g., Wu et al. 2017) considers the joint lot-sizing and cutting stock problem, where components are cut from sheets of different sizes.

For the multi-echelon case, Begnaud et al. (2009) consider the case of a flexible BOM, where each item can be produced with a different recipe. Each recipe corresponds to a different task in the BOM, and the model selects one task among all possible ones. A variant of the problem is called multi-echelon lot-sizing with BOM substitution (Wei et al. 2019). In this variant, the entire BOM can be modified in each period, and all items use the same BOM in a period (but different BOMs can be used in different periods).

To the best of our knowledge, lot-sizing with component substitution has never been considered in a stochastic demand situation. However, there exists a wide literature (e.g., Gallego et al. 2006, Han et al. 2014) on inventory systems with product substitution (Shin et al. 2015). This stream of research differs from the current study since the substitution concerns the end-items and not the components. Product substitution can be customer-driven if the customer selects a different product when the one he wanted is not available, or supplier-driven if the supplier decides to downgrade a product to meet the demand (Huang et al. 2011). There exist several studies considering stochastic optimization for inventory systems with product substitution. For instance, Hsu and Bassok (1999) propose a scenario-based stochastic optimization approach for a production system under random yield and random demand with product substitution. To speed up the solution process, the authors rely on Benders decomposition, and they show that a greedy algorithm can solve the sub-problems since

they have a special network flow. Also, Rao et al. (2004) propose a stochastic optimization method and heuristics for the single-period inventory system with setup in a static-dynamic framework. The authors show that substitution results in substantial savings, especially when the setup costs or the demand variance are large.

## 2.2 Literature review on lot-sizing under stochastic demand

Compared to the vast literature on deterministic lot-sizing, only a limited number of works consider the stochastic version of this problem. There exist various sources of uncertainty in lot-sizing, such as demand, yield, lead times, or capacity, and the present work focuses on demand uncertainty. As meeting the demand in any situation is expensive, stochastic optimization models seek to minimize the expected backlog (e.g., Tarim and Kingsman 2006, Tunc et al. 2018) or to reach a given service level (e.g., Bookbinder and Tan 1988) with chance constraint approaches. In addition, three decision frameworks exist, namely, static-static, static-dynamic, and dynamic-dynamic. In static-static, the lot sizes are determined in period 0 for the entire horizon, and they are frozen. On the contrary, in dynamic-dynamic, the decisions are made in each period, after having observed the realization of the stochastic parameters. In static-dynamic, the setups are static (selected in the first period for the entire horizon and frozen), whereas the quantities are computed in each period after the unknown parameters unfold. For more information on stochastic lot-sizing, the interested reader is referred to the recent surveys of Tempelmeier (2013) and of Aloulou et al. (2014).

Few works consider the capacitated lot-sizing problem with stochastic demands (e.g., Tempelmeier and Hilger 2015, Brandimarte 2006). However, most studies on stochastic lot-sizing consider a single-echelon production system. To the best of our knowledge, the only papers on stochastic optimization for multi-echelon systems are Grubbström and Wang (2003), Quezada et al. (2020), and Thevenin et al. (2020). Grubbström and Wang (2003) propose a dynamic programming approach to minimize the net present value in the capacitated multi-echelon lot-sizing problem with stochastic demand, but they ignore lead times. Quezada et al. (2020) propose a branch-and-cut approach for the uncapacitated lot-sizing problem in a re-manufacturing system with three echelons under various types of uncertainty (demand, cost, supply, yield). Thevenin et al. (2020) propose a scenario tree approach for the multi-echelon capacitated lot-sizing problem under stochastic demand. Thevenin et al. (2020) show that stochastic optimization significantly reduces the costs in material requirements planning when compared to traditional approaches used in lot-size and safety stock determination. In the present work, we propose a stochastic dual dynamic programming approach to solve requirements planning with component substitution over a long planning horizon. To alleviate the scalability issues, we develop a heuristic version of SDDP, and this method can handle a much larger set of scenarios than the MIP proposed in Thevenin et al. (2020).

Optimization methods commonly used for stochastic lot-sizing include progressive hedging (Haugen et al. 2001), scenario based mathematical programming (Brandimarte 2006), dynamic programming (Sox 1997), heuristics (Piperagkas et al. 2012), branch-and-bound (Tarim et al. 2011), among others. However, to the best of our knowledge, this paper is the first to propose a SDDP algorithm for a lot-sizing problem. The SDDP algorithm has been mostly applied to hydro-thermal energy operations planning (e.g., Soares et al. 2017, Lohmann et al. 2016), and other applications include the control of microgrids (Pacaud et al. 2018), pastoral dairy farms (Dowson et al. 2019), or portfolio optimization (Valladão et al. 2019). The multi-echelon lot-sizing problem under demand uncertainy requires a large number of scenarios to approximate the costs (Thevenin et al. 2020). Consequently, the classical scenario tree approach leads to a huge Mixed Integer Linear Program (MILP) that consumes too much memory. In this context, decomposition approaches such as PH (decomposition per scenario) and SDDP (decomposition per stage) are suitable and could be highly efficient in addressing the tractability issue in this problem. While existing studies (e.g., Brandimarte 2006, Quezada et al. 2020) on scenario tree methods for lot-sizing are limited to less than 1000 scenarios, the SDDP algorithm proposed in the present work computes the production quantities based on $20^{10}$ scenarios.

Our contributions are threefold. (1) This paper is the first to study lot-sizing with component substitution under stochastic demand. Our analysis shows that component substitution can pool the risk, and it allows maintaining the same service level with less inventory. (2) We investigate the performance of SDDP for the lot-sizing problem, and we study the impact of multiple algorithmic improvements of the SDDP. (3) We propose two strategies to compute and fix the setups in the SDDP framework, namely, a hybrid of PH and SDDP and a heuristic version of SDDP. Our results show that the heuristic is competitive with the MILP on small scenario trees, but it can handle much larger scenario sets.

# 3 Considered problem

This paper addresses the capacitated multi-echelon lot-sizing problem with component substitution (CMLCS) under uncertain demand. Given a multi-echelon bill of materials, the demand distribution of the end-items, and the unit capacity consumption of each item, the CMLCS is to decide when to produce as well as the sizes of the production lots, and to select the components to minimize the expected setup, production, substitution, inventory, backlog, and lost sales costs.

The BOM gives the production recipe of each item in the set $I_e$ of end-items. More precisely, the production of item $j$ consumes $R_{ij}$ units of component $i$. However, an item $i'$ in the set $\mathcal{A}_i$ can substitute component $i$, and this substitution costs $a_{i'i}$. For simplicity, we assume $i \in \mathcal{A}_i$ with $a_{ii} = 0$. We denote by $\mathcal{I}_c$ the set of components and by $\mathcal{I}$ the set of all items with $\mathcal{I} = \mathcal{I}_e \cup \mathcal{I}_c$. Without loss of generality, we assume that there is no demand for components, whereas the demand's probability distribution of each end-item is given for each period in the planning horizon $\mathcal{H} = \{1, \dots, T\}$. These uncertain demands are represented by the set $\Omega$ of demand scenarios, where each scenario $\omega \in \Omega$ has a probability $p_\omega$. The requirement plan must account for the production capacity and lead times. Each resource $r$ in the set of resources $\mathcal{R}$ has a given capacity $C_r$, and the production of one unit of item $i$ consumes $K_{ir}$ units of capacity of resource $r$. The item $i$ produced in period $t$ are available in period $t + L_i$. The objective function is the expected total cost, and it includes the backlog cost $b_i$, holding cost $h_i$, production cost $v_i$, lost sale cost $e_i$ per unit of item $i$, as well as the the setup cost $s_i$ for each lot of item $i$, and the substitution costs $a_{i'i}$ per unit.

In the static-dynamic type of uncertainty, the setups $Y_{it}$ for each item $i$ are decided in period 0 for all the periods $t$ in the horizon, and they are frozen. On the contrary, the inventory $I^\omega_{it}$ and backlog level $B^\omega_{it}$ are computed after observing the demand in each scenario $\omega$, and the quantities $Q^\omega_{it}$ for each item $i$ ordered in period $t$ are computed after having observed the demand in period $t-1$ in each scenario $\omega$. Similarly, the quantity $W^\omega_{ijt}$ of component $i$ consumed to meet the internal demand of item $j$ in period $t$ is decided after observing the demand in period $t-1$ for each scenario $\omega$. Note that the backlog level $B^\omega_{iT}$ in period $T$ corresponds to a lost sale.

The mathematical formulation of CMLCS is the following:

$$\min \sum_{\omega\in\Omega} p_\omega \left( \sum_{t\in\mathcal{H}} \sum_{i\in\mathcal{I}} \left( h_i I^\omega_{it} + s_i Y_{it} + v_i Q^\omega_{it} + \sum_{j\in\mathcal{I}} a_{ij} W^\omega_{ijt} \right) + \sum_{i\in\mathcal{I}_e} \left( \sum_{t=1}^{t=T-1} b_i B^\omega_{it} + e_i B^\omega_{iT} \right) \right) \tag{1}$$

$$\text{s.t.} \quad I^\omega_{it-1} - B^\omega_{it-1} + Q^\omega_{it-L_i} - D^\omega_{it} - I^\omega_{it} + B^\omega_{it} = 0 \qquad i \in \mathcal{I}_e,\ t \in \mathcal{H},\ \omega \in \Omega \tag{2}$$

$$I^\omega_{it-1} + Q^\omega_{it-L_i} - \sum_{j\in\mathcal{I}} W^\omega_{ijt} - I^\omega_{it} = 0 \qquad i \in \mathcal{I}_c,\ t \in \mathcal{H},\ \omega \in \Omega \tag{3}$$

$$\sum_{i\in\mathcal{A}_k} W^\omega_{ikt} = \sum_{j\in\mathcal{I}} R_{kj} \cdot Q^\omega_{jt} \qquad k \in \mathcal{I}_c,\ t \in \mathcal{H},\ \omega \in \Omega \tag{4}$$

$$Q_{it} \le M_i Y_{it} \qquad i \in \mathcal{I},\ t \in \mathcal{H} \tag{5}$$

$$\sum_{i\in\mathcal{I}} K_{ir} Q^\omega_{it} \le C_r \qquad t \in \mathcal{H},\ r \in \mathcal{R},\ \omega \in \Omega \tag{6}$$

$$W^{\omega}_{ijt+1} = W^{\omega'}_{ijt+1},\ Q^{\omega}_{it+1} = Q^{\omega'}_{it+1} \qquad i \in \mathcal{I},\ t \in \mathcal{H},\ \omega, \omega' | D^{\omega}_{\mathcal{I}_e 1 \dots t} = D^{\omega'}_{\mathcal{I}_e 1 \dots t} \tag{7}$$

$$B^{\omega}_{it} \geq 0 \qquad i \in \mathcal{I}_e,\ t \in \mathcal{H},\ \omega \in \Omega \tag{8}$$

$$I^{\omega}_{it} \geq 0 \qquad i \in \mathcal{I},\ t \in \mathcal{H},\ \omega \in \Omega \tag{9}$$

$$W^{\omega}_{ijt} \geq 0 \qquad i,\ j \in \mathcal{I},\ t \in \mathcal{H},\ \omega \in \Omega \tag{10}$$

$$Q^{\omega}_{it} \geq 0 \qquad i \in \mathcal{I},\ t \in \mathcal{H},\ \omega \in \Omega \tag{11}$$

$$Y_{it} \in \{0, 1\} \qquad i \in \mathcal{I},\ t \in \mathcal{H}. \tag{12}$$

The objective function (1) is the expected sum of inventory costs, setup costs, unit production costs, backlog costs, end-of-horizon lost sales costs, and component substitution costs. The backlog and inventory quantities are computed with constraints (2) for end-items and constraints (3) for components. Constraints (4) compute the component consumption associated with the production quantities. Constraints (5) set the production quantities to zero in periods without setups, where the value of $M_i$ is the minimum between the lower bound $M^1_i$ on the production quantities computed from the demands (13) and the lower bound $M^2_i$ computed from the resource capacities (14). Finally, constraints (6) and (7) are the capacity and non-anticipativity constraints, respectively. In (7), $D^{\omega}_{\mathcal{I}_e 1 \dots t}$ refers to the demand matrix $D^{\omega}_{it}$ for each item $i \in \mathcal{I}_e$ and period $t \in 1 \dots t$

The bounds $M^1_i$ and $M^2_i$ are computed as follows:

$$M^1_i = \begin{cases} \max_{\omega \in \Omega} \sum_{t \in \mathcal{H}} D^{\omega}_{it} & \text{if } i \in \mathcal{I}_e \\ \sum_{j \in \mathcal{I}} \left( \sum_{k | i \in \mathcal{A}_k} R_{kj} \right) M^1_j & \text{if } i \in \mathcal{I}_c \end{cases} \tag{13}$$

$$M^2_i = \min_{k \in \mathcal{K} | K_{ik} > 0} \frac{C_k}{K_{ik}} \tag{14}$$

# 4 Stochastic Dual Dynamic Programming for the CMLCS

This section describes the proposed SDDP algorithm for the CMLCS. Section 4.1 gives a generic description of SDDP, and Section 4.2 describes its adaptation to the CMLCS. Finally, Section 4.3 presents the enhancements for SDDP.

## 4.1 The Stochastic Dual Dynamic Programming framework

To avoid the exponential growth in the number of states, SDDP assumes that the probability distributions are stage-wise independent. Under this assumption, the complete set of scenarios $\Omega$ corresponds to a symmetric scenario tree, where each node represents a realization of the stochastic parameters, and each path is a scenario. Figure 1 shows such a tree for the single end-item CMLCS, with the realization of the demand on each node, and the probability associated with the realization on the edge leading to each node. As the tree is symmetric, all the nodes of stage $t$ have branches toward the same scenario set $\Omega_{t+1}$ for the parameters revealed in stage $t+1$, and the branches' probability toward the same sample are identical. Consequently, the stochastic process can be represented with a sample $\Omega_t$ for each period $t \in T$.

SDDP decomposes the stochastic optimization problem per decision stage. Given the decisions $x_{1 \dots t-1}$ made in stages 1 to $t-1$ and the realization $\omega_{1 \dots t}$ of the stochastic parameters in stages 1 to $t$, the sub-problem of stage $t$ is to find the decisions $x_t$ optimizing the costs of stage $t$ plus the cost-to-go $\mathcal{F}_{t+1}(\omega_{1 \dots t}, x_{1 \dots t})$. This cost-to-go function represents the expected costs from stage $t+1$ to $T$ as a function of the decisions $x_{1 \dots t}$ made from stage 1 to $t$ and the realization $\omega_{1 \dots t}$ of the stochastic parameters in stages 1 to $t$. As the cost-to-go functions are unknown at the start, SDDP iteratively builds an outer approximation of these functions. The reader is referred to Pereira and Pinto (1991) and Shapiro (2011) for more information about SDDP.

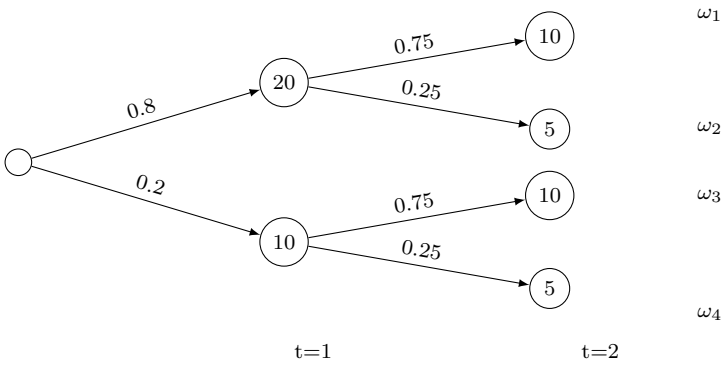


**Figure 1: Example of symmetric scenario tree [2, 2]**

## 4.2 Stochastic Dual Dynamic Programming for CMLCS

At each iteration of SDDP, a forward pass generates a feasible solution, and a backward pass adds information about the cost-to-go functions. The forward and backward steps are described below, and they are followed by a discussion on the stopping criterion.

### 4.2.1 Forward pass

The forward pass simulates the use of the current policy for a set $\Xi$ of scenarios re-sampled from $\Omega$ at each iteration. For each scenario $\xi \in \Xi$, the sub-problems are solved starting from stage $t = 1$ until stage $t = T$. Each decision stage corresponds to the arrival of new information on the end-items demands. More precisely, the first stage corresponds to the initial state, where no information on the demand is available, and it decides the quantities to produce in period 1. Similarly, the stage $t \in \{2, \dots, T\}$ corresponds to the beginning of period $t$, where the demands of end-items in period $t-1$ are known, and it decides the quantities to produce in period $t$. Finally, the last stage computes the inventory and backlog costs when the last period demands are known. This section successively presents the sub-problem in stage 1, the sub-problem in stage $t \in \{2, \dots, T\}$, and the sub-problem in the last stage. For ease of exposition, we assume strictly positive lead times for all items, but the proposed algorithm can easily be extended to the case of zero lead times. For clarity, we denote by $Q_{\mathcal{I}\mathcal{H}}$ the matrix of quantity variables $Q_{it}$ for all $i \in \mathcal{I}$ and $t \in \mathcal{H}$, and the same notation applies to other variables. Besides, to clarify the stage-wise decomposition, we refer to the past period with scenario $\xi$ and the present scenario as $\omega$, but in the forward pass $\xi = \omega$ since it simulates the decisions for each scenario.

**First stage model.** The first stage decisions are made when the demand in periods 1 to $t$ is unknown. These decisions include the setups $Y_{it}$ for all items $i$ and periods $t$, the production quantities $Q_{i1}$ for all items $i$ in period 1, the consumed quantities $W_{ij1}$ for all arcs $(i, j)$ in the BOM in period 1, and the inventory levels $I_{i1}$ of components in period 1. Contrarily to the components inventory, the end-item inventory levels in period 1 depend on the realization of the demand in period 1, and they are thus decided in stage 2. Note that the mathematical model includes a variable $\mathcal{F}_2$ specifying the lower bound of the cost-to-go. This cost-to-go function is defined by a set $\mathcal{G}_2$ of hyper planes built iteratively to improve the value of $\mathcal{F}_2$. The first stage problem can be formulated as follows:

$$\text{Min} \quad \sum_{i \in \mathcal{I}} \left( \sum_{t \in \mathcal{H}} s_i Y_{it} + v_i Q_{i1} + \sum_{j \in \mathcal{I}} a_{ij} W_{ij1} \right) + \sum_{i \in \mathcal{I}_c} h_i I_{i1} + \mathcal{F}_2 \tag{15}$$

$$\text{s.t.} \quad I_{i0} - \sum_{j \in \mathcal{I}} W_{ij1} - I_{i1} = 0 \qquad i \in \mathcal{I}_c \tag{16}$$

$$\sum_{i \in \mathcal{A}_k} W_{ik1} = \sum_{j \in \mathcal{I}} R_{kj} Q_{j1} \qquad k \in \mathcal{I}_c,\ j \in \mathcal{I} \tag{17}$$

$$Q_{i1} \leq Y_{i1} M \qquad i \in \mathcal{I} \tag{18}$$
$$\sum_{i \in \mathcal{I}} Q_{i1} K_{ir} \leq C_r \qquad r \in \mathcal{R} \tag{19}$$
$$\mathcal{F}_2 \geq \mathcal{F}_2^h(Y_{\mathcal{I}\mathcal{H}}, Q_{\mathcal{I}1}, I_{\mathcal{I}_c 1}) \qquad h \in \mathcal{G}_2 \tag{20}$$
$$Q_{i1} \geq 0 \qquad i \in \mathcal{I} \tag{21}$$
$$W_{ij1} \geq 0 \qquad i \in \mathcal{I},\ j \in \mathcal{I} \tag{22}$$
$$I_{i1} \geq 0 \qquad i \in \mathcal{I}_c \tag{23}$$
$$Y_{it} \in \{0,1\} \qquad i \in \mathcal{I},\ t \in \mathcal{H} \tag{24}$$
$$\mathcal{F}_2 \geq 0. \tag{25}$$

The objective function (15) includes the setup costs for all products and all periods, the production and consumption costs for all items in period 1, the inventory costs of components in period 1, and the approximation of the future costs $\mathcal{F}_2$. Constraints (16) compute the inventory levels of components, and constraints (17) determine the consumed quantities. Constraints (18) set the quantity to 0 if there is no setup, and the capacity constraints are given in (19). Finally, the future costs (20) is defined by all hyper planes $\mathcal{F}_2^h(Y_{\mathcal{I}\mathcal{H}}, Q_{\mathcal{I}1}, I_{\mathcal{I}_c 1})$ in the set $\mathcal{G}_2$.

**Sub-problem of stage $t \in \{2,\dots,T\}$.** In stage $t \in \{2,\dots,T\}$, the end-item demand of period $t-1$ unfolds, and the inventory level $I^{\omega}_{it-1}$ and backlog $B^{\omega}_{it-1}$ at the end of period $t-1$ can be computed for the end-item $i$ in scenario $\omega$. Based one this information, the sub-problem of stage $t$ computes the quantities $Q^{\omega}_{it}$ and $W^{\omega}_{ijt}$. Similarly to the stage 1 sub-problem, the inventory level $I^{\omega}_{it}$ in period $t$ for components $i$ in scenario $\omega$ depends on the internal demand, and it can be computed in stage $t$. The decisions are chosen to minimize the costs given the state of the system at the beginning of stage $t$. The state of the system comes from the solution of the forward pass for scenario $\xi$ in previous stages. This state can be described with the initial inventory $\widehat{I}^{\xi}_{it-2}$ and $\widehat{B}^{\xi}_{it-2}$ for end-items and $\widehat{I}^{\xi}_{it-1}$ for components, as well as with the quantities $\widehat{Q}^{\xi}_{it-L_i}$ and $\widehat{Q}^{\xi}_{it-L_i-1}$ received in period $t-1$. The formulation of the stage $t \in \{2,\dots,T\}$ sub-problem is given as follows:

$$\mathcal{F}_t(\widehat{Y}_{\mathcal{I}t}, \widehat{Q}^{\xi}_{\mathcal{I}_e t-L_i-1}, \widehat{Q}^{\xi}_{\mathcal{I}_c t-L_i}, \widehat{I}^{\xi}_{\mathcal{I}_e t-2}, \widehat{I}^{\xi}_{\mathcal{I}_c t-1}, \widehat{B}^{\xi}_{\mathcal{I}_e t-2}, D^{\omega}_{\mathcal{I}_e t-1}) =$$

$$\text{Min} \sum_{i \in \mathcal{I}} v_i Q^{\omega}_{it} + \sum_{i \in \mathcal{I}_c} h_i I^{\omega}_{it} + \sum_{i \in \mathcal{I}_e} \left(b_i B^{\omega}_{it-1} + I^{\omega}_{it-1}\right) + \sum_{i \in \mathcal{I}} \sum_{j \in \mathcal{I}} a_{ij} W^{\omega}_{ijt} + \mathcal{F}_{t+1} \tag{26}$$
$$\text{s.t. } \widehat{Q}^{\xi}_{it-L_i-1} + \widehat{I}^{\xi}_{it-2} - \widehat{B}^{\xi}_{it-2} - D^{\omega}_{it-1} - I^{\omega}_{it-1} + B^{\omega}_{it-1} = 0 \qquad i \in \mathcal{I}_e \tag{27}$$
$$\widehat{Q}^{\xi}_{it-L_i} + \widehat{I}^{\xi}_{it-1} - \sum_{j \in \mathcal{I}} W_{ijt} - I^{\omega}_{it} = 0 \qquad i \in \mathcal{I}_c \tag{28}$$
$$\sum_{i \in \mathcal{A}_k} W^{\omega}_{ikt} = \sum_{j \in \mathcal{I}} R_{kj} Q^{\omega}_{jt} \qquad k \in \mathcal{I}_c,\ j \in \mathcal{I} \tag{29}$$
$$Q^{\omega}_{it} \leq M_i \widehat{Y}_{it} \qquad i \in \mathcal{I} \tag{30}$$
$$\sum_{i \in \mathcal{I}} Q^{\omega}_{it} K_{ir} \leq C_r \qquad r \in \mathcal{R} \tag{31}$$
$$\mathcal{F}_{t+1} \geq \mathcal{F}^h_{t+1}\big(\widehat{Y}_{\mathcal{I}\mathcal{H}}, \widehat{Q}^{\xi}_{\mathcal{I}_e 1\dots t-2}, Q^{\omega}_{\mathcal{I}_e t-1}, \widehat{Q}^{\xi}_{\mathcal{I}_c 1\dots t-1}, Q^{\omega}_{\mathcal{I}_c t-1}, \widehat{I}^{\xi}_{\mathcal{I}_e 1\dots t-2}, I^{\omega}_{\mathcal{I}_e t-1}, \widehat{I}^{\xi}_{\mathcal{I}_c 1\dots t-1}, I^{\omega}_{\mathcal{I}_c t}, \widehat{B}^{\xi}_{\mathcal{I}_e 1\dots t-2}, B^{\omega}_{it-1}, \widehat{B}^{\xi}_{\mathcal{I}_c 1\dots t-1}, B^{\omega}_{\mathcal{I}_c t}\big) \qquad h \in \mathcal{G}_{t+1} \tag{32}$$
$$B^{\omega}_{it-1}, I^{\omega}_{it-1}, Q^{\omega}_{it} \geq 0 \qquad i \in \mathcal{I}_e \tag{33}$$
$$W^{\omega}_{ijt}, \geq 0 \qquad i \in \mathcal{I},\ j \in \mathcal{I} \tag{34}$$
$$I^{\omega}_{it}, Q^{\omega}_{it} \geq 0 \qquad i \in \mathcal{I}_c \tag{35}$$
$$\mathcal{F}_{t+1} \geq 0. \tag{36}$$

The objective function (26) includes the production costs, the inventory costs of the components in period $t$, the backlog and inventory costs of end-items in period $t-1$, the component substitution costs, and the approximation of the future costs $\mathcal{F}_{t+1}$. Constraints (27) and (28) compute the inventory and backlog levels for end-items and components, respectively. Finally, constraints (29), (30) and (31) are the consumption, setup and capacity constraints, respectively. Similarly to the first stage model, the cost-to-go function is defined by the set of hyper planes $\mathcal{G}_{t+1}$ in (32).

**Last stage sub-problem.** The last stage $(T+1)$ sub-problem is given below, where the decision variables include the end-item inventory $I^{\omega}_{iT}$ and the lost sales $B^{\omega}_{iT}$ for end-item $i$ in period $T$ for scenario $\omega$:

$$\mathcal{F}_{T+1}(\widehat{Q}^{\xi}_{\mathcal{I}_e T-L_i}, \widehat{I}^{\xi}_{\mathcal{I}_e T-1}, \widehat{B}^{\xi}_{\mathcal{I}_e T-1}) =$$

$$\text{Min} \sum_{i\in\mathcal{I}_e} (h_i \cdot I^{\omega}_{iT} + e_i B^{\omega}_{iT}) \tag{37}$$

$$\text{s.t. } \widehat{Q}^{\xi}_{iT-L_i} + \widehat{I}^{\xi}_{iT-1} - \widehat{B}^{\xi}_{iT-1} - D^{\omega}_{iT} - I^{\omega}_{iT} + B^{\omega}_{iT} = 0 \qquad i \in \mathcal{I}_e \tag{38}$$

$$I^{\omega}_{iT}, B^{\omega}_{iT} \geq 0 \qquad i \in \mathcal{I}_e. \tag{39}$$

The objective (37) includes the lost sales and inventory costs of end-items, and their values are computed with constraints (38).

### 4.2.2 Backward pass

The outer approximation of the cost-to-go $\mathcal{F}_{t+1}(\omega_{1\ldots t}, x_{1\ldots t-1})$ is a set of hyper planes. This set is initially empty, and it is built during the *backward pass*. To compute the cost-to-go associated with the scenarios in $\Omega_t$ given the decisions made in the previous stages, the backward pass solves the sub-problem of each stage $t$ (from $t = T$ to $t = 2$) for all scenarios $\omega$ in $\Omega_t$. To improve the approximation of the cost-to-go in stage $t-1$, SDDP adds a cut inferred from the dual of the stage $t$ sub-problems (i.e., Benders optimality cuts) to the sub-problem of stage $t-1$.

**Backward pass in stage $T+1$.** The dual of sub-problem (37)–(39) is:

$$\text{Max} \sum_{\omega\in\Omega} p_\omega \sum_{i\in\mathcal{I}_e} \lambda^{k\omega}_i \left(-\widehat{Q}^{\xi}_{iT-L_i} - \widehat{I}^{\xi}_{iT-1} + \widehat{B}^{\xi}_{iT-1} + D^{\omega}_{iT}\right) \tag{40}$$

$$\text{s.t. } -\lambda^{k\omega}_i \leq h_i \qquad i \in \mathcal{I}_e \tag{41}$$

$$\lambda^{k\omega}_i \leq e_i \qquad i \in \mathcal{I}_e. \tag{42}$$

Consequently, the hyper plane

$$\sum_{\omega\in\Omega} p_\omega \sum_{i\in\mathcal{I}_e} \lambda^{k\omega}_i \left(-\widehat{Q}^{\xi}_{iT-L_i} - I^{\omega}_{iT-1} + B^{\omega}_{iT-1} + D^{\omega}_{iT}\right) \tag{43}$$

is added to $\mathcal{G}_T$, where $\lambda^{k\omega}_i$ is the dual variable of constraint (38) for product $i$ in scenario $\omega$ at iteration $k$. Note that $I^{\omega}_{iT-1}$ and $B^{\omega}_{iT-1}$ are the variables of stage $T$, whereas the variables $\widehat{Q}^{\omega}_{iT-L_i}$ are fixed during the forward pass at stage $T-L_i$.

**Backward pass at stage $t \in \{2,\ldots,T\}$.** The cuts generated from the sub-problems in stage $t \in \{2,\ldots,T\}$ are also based on the objective of the dual. In the first iteration of the algorithm, the dual

of the sub-problem in stage $t$ is

$$\sum_{\omega\in\Omega} p_\omega \Bigg( \sum_{i\in\mathcal{I}_e} \kappa_i^{k\omega} \Big( - \widehat{Q}^{\xi}_{it-L_i-1} - I^{\omega}_{it-2} + B^{\omega}_{it-2} + D^{\omega}_{it-1} \Big) + \sum_{i\in\mathcal{I}_c} \eta_i^{k\omega} \Big( - \widehat{Q}^{\xi}_{it-L_i} - I^{\omega}_{it-1} \Big) + \sum_{i\in\mathcal{I}} \theta_i^{k\omega} M \widehat{Y}_{it} + \sum_{r\in\mathcal{K}} \chi_i^{k\omega} C_r \Bigg),$$

where $\kappa_i^{k\omega}$, $\eta_i^{k\omega}$, $\theta_i^{k\omega}$, $\chi_i^{k\omega}$ denote, respectively, the dual variables of constraints (27), (28), (30), (31) for item $i$ in scenario $\omega$ in iteration $k$.

However, SDDP iteratively generates a set of cuts in the sub-problems, and these cuts modify the dual's objective. For instance, at stage $T$, the cut (43) is added in each iteration, and the following element is added to the dual's objective:

$$\sum_{h\in\mathcal{G}_T} \pi_h \sum_{\omega\in\Omega} p_\omega \sum_{i\in\mathcal{I}_e} \lambda_i^{h\omega} (-\widehat{Q}^{\omega}_{iT-L_i} + D^{\omega}_{it}),$$

where $\mathcal{G}_T$ denotes the set of cuts in stage $T$, and $\pi_h$ denotes the dual variables of cut $h$ (added in iteration $h$). Thus, after solving the sub-problem of stage $T$, the following cut is added to the sub-problem of stage $T-1$:

$$\mathcal{F}_T \geq \sum_{\omega\in\Omega} p_\omega \Big( \sum_{i\in\mathcal{I}_e} \kappa_i^{k\omega} (-\widehat{Q}^{\xi}_{iT-L_i-2} - I^{\omega}_{it-3} + B^{\omega}_{iT-3} + D^{\omega}_{iT-2} \Big) + \sum_{i\in\mathcal{I}_c} \eta_i^{k\omega} (-\widehat{Q}^{\xi}_{iT-L_i-1} - I^{\omega}_{iT-2}) + \sum_{i\in\mathcal{I}} \theta_i^{k\omega} M \widehat{Y}_{iT-1} + \sum_{r\in\mathcal{K}} \chi_i^{k\omega} C_r + \sum_{h\in\mathcal{G}_T} \pi_h \sum_{i\in\mathcal{I}_e} \lambda_i^{h\omega} (-\widehat{Q}^{\omega}_{iT-L_i} + D^{\omega}_{it}) \Big).$$

Note that if $L_i = 1$, $\widehat{Q}^{\omega}_{iT-L_i}$ is a variable of the stage $T-1$ sub-problem, whereas for $L_i > 1$, $\widehat{Q}^{\omega}_{iT-L_i}$ is fixed to the value found in the previous stage.

We explain below the construction of the cut for any stage $t > T-1$. Each cut $c$ in the set of cuts $\mathcal{C}_t$ of stage $t$ can be expressed by

$$\mathcal{F}_{t+1} \geq B^c_{t-1} \widehat{X}_{t-1} + B^c_t X_t + B^c_e \widehat{X}_e + b^c,$$

where $X_t$ denotes the variables of stage $t$, $\widehat{X}_{t-1}$ is the set of variables of stage $t-1$, $\widehat{X}_e$ is the set of variables decided earlier than stage $t-1$ , $B^c_{t-1}$, $B^c_t$, $B^c_e$ are the matrix of coefficients, and $b^c$ is a constant. These cuts increase the objective function value of the dual with

$$\sum_{h\in\mathcal{G}_t} \pi_c \left( B^c_{t-1} X_{t-1} + B^c_e \cdot \widehat{X}_e + b^c \right),$$

where $\pi_c$ is the dual variable associated with cut $c$. Consequently, after re-solving the sub-problem at stage $t$, the following cut is added to the sub-problem of stage $t-1$:

$$\begin{aligned}\mathcal{F}_t \geq \sum_{\omega\in\Omega} p_\omega \Big( & \sum_{i\in\mathcal{I}_e} \kappa_i^{k\omega}(-\widehat{Q}^{\xi}_{it-L_i-1} - I^{\omega}_{it-2} + B^{\omega}_{it-2} + D^{\omega}_{it-1}\Big) \\ & + \sum_{i\in\mathcal{I}_c} \eta_i^{k\omega}(-\widehat{Q}^{\xi}_{it-L_i} - I^{\omega}_{it-1}) \\ & + \sum_{i\in\mathcal{I}} \theta_i^{k\omega} M\widehat{Y}_{it} \\ & + \sum_{r\in\mathcal{K}} \chi_i^{k\omega} C_r \\ & + \sum_{h\in\mathcal{G}_t} \pi_c \left(B^c_{t-1}X_{t-1} + B^c_e\cdot\widehat{X}_e + b^c\right)\Big).\end{aligned}$$

The Algorithms 1, 2, and 3 give the steps of SDDP, the forward pass, and the backward pass, respectively.

**Algorithm 1** SDDP

Generate the first stage MILP, and the LP of the subsequent stages.
**while** the stopping criterion is not met **do**
  Sample a set of scenarios $\Xi$.
  **for** each scenario $\xi \in \Xi$ **do**
    Perform the forward pass (*Algorithm 2*) to find the solution for scenario $\xi$ with the current policy.
    Perform the backward pass (*Algorithm 3*) to improve the approximation of the cost-to-go.
  **end for**
**end while**

**Algorithm 2** Forward pass on a scenario $\xi$

Solve the first stage MIP (15) - (25) to get the values of $\widehat{Y}_{\mathcal{I}t}$, and $\widehat{Q}_{\mathcal{I}1}$
**for** $t = 2$ to $t = T$ **do**
  Compute $\mathcal{F}_t(\widehat{Y}_{\mathcal{I}t}, \widehat{Q}^{\xi}_{\mathcal{I}_e t-L_i-1}, \widehat{Q}^{\xi}_{\mathcal{I}_c t-L_i}, \widehat{I}^{\xi}_{\mathcal{I}_e t-2}, \widehat{I}^{\xi}_{\mathcal{I}_c t-1}, \widehat{B}^{\xi}_{\mathcal{I}_e t-2})$ and record the resulting values of $\widehat{Q}_{\mathcal{I}t}$, $\widehat{I}_{\mathcal{I}_c t}$, $\widehat{B}_{\mathcal{I}_c t}$, $\widehat{I}_{\mathcal{I}_e t-1}$, $\widehat{B}_{\mathcal{I}_e t-1}$.
**end for**

**Algorithm 3** Backward pass

**for** $t = T+1$ to $t = 2$ **do**
  **for** each scenario $\omega \in \Omega$ **do**
    Modify the LP (26) - (36) of stage $t$ according to the demand of scenario $\omega$ and the decisions made in previous stages 1 to $t-1$ of the forward pass, and solve this LP.
  **end for**
  Create a new cut for stage $t-1$ as shown in Equation (44).
**end for**

### 4.2.3 Stopping condition

On the one hand, as $\mathcal{F}_2$ is a partial outer-approximation of the cost-to-go, the solution of the first stage sub-problem gives a lower bound $LB$ of the optimal solution. On the other hand, the forward pass gives an approximate upper bound $UB$, since the forward pass gives the expected cost of the current policy over the subset $\Xi$ of scenarios (the resulting solutions are feasible but the true cost is not calculated unless all the scenarios are evaluated). However, the convergence of SDDP cannot be detected by a simple comparison of $UB$ and $LB$, because $UB$ is only an approximate upper bound. Therefore, in this work, we stop the algorithm when a predetermined time limit is reached.

## 4.3 Enhancements to SDDP

To speed up the convergence of SDDP, several enhancements are proposed, namely, the generation of strong cuts, retaining the average scenario, generating multiple cuts in the backward pass, and advanced scenario sampling. These strategies are described below.

### 4.3.1 Generation of strong cuts

Magnanti and Wong (1981) observed that the dual of a degenerate sub-problem may have multiple optimal solutions, and that each solution leads to the generation of a different Benders optimality cut. Therefore, the authors proposed to accelerate Benders decomposition by generating the strongest cuts. Later, Papadakos (2008) proposed a practical enhancement called the interior point method.

To solve the sub-problem of stage $t$ in the backward pass, the previous stage variables are not fixed to the forward pass decisions, but to the values of an approximate core point. In the first iteration, the approximate core point $Y^C_{it}$, $Q^C_{it-L_i}$, $I^C_{it-1}$, and $B^C_{it-1}$ is set to the forward pass solution. Next, in each iteration $k$, the approximate core point is updated with the forward pass solution ($Y^k_{it}$,$Q^k_{it}$, $I^k_{it}$ ) of iteration $k$ as follows:

$$\begin{aligned} Y^C_{it} &= pY^C_{it} + (1-p)Y^k_{it} \\ Q^C_{it-L_i} &= pQ^C_{it-L_i} + (1-p)Q^k_{it-L_i} \\ I^C_{it-1} &= pI^C_{it-1} + (1-p)I^k_{it-1} \\ B^C_{it-1} &= pB^C_{it-1} + (1-p)B^k_{it-1}, \end{aligned}$$

where $p < 1$ is a parameter (set to 0.5 in our implementation). Note that this implementation assumes $|\Xi| = 1$. The strong cuts are common in the literature on Benders decomposition but are not often used in SDDP. Nevertheless, they remain valid, and they are easy to implement.

### 4.3.2 Lower bound inequality from the expected demand problem

As SDDP builds iteratively the approximation of the cost-to-go, this approximation is not precise in the first iterations. However, the computation of the cost-to-go based on the expected demand scenario can help drive SDDP towards good solutions despite this inaccuracy.

Given a stochastic optimization program $P$ such that the coefficients of the variables in the objective function and in the constraints are constant, Birge and Louveaux (2011) show that

$$EV \leq f,$$

where $EV$ is the cost of problem P solved with the scenario of the expected demand, and $f$ is the value of the optimal solution of P. Consequently, the expected cost-to-go $\mathcal{F}_{t+1}$ from period $t+1$ to $T$ is larger than or equal to the cost-to-go computed with the average demand scenario.

To drive SDDP towards good solutions during the first few iterations, this lower bound is computed in each sub-problem. As the lower bound is computed by retaining the average scenario in the first stage problem, this approach is similar to the partial Benders Decomposition proposed by Crainic et al. (2016). More precisely, the sub-problems include the variables $\bar{Q}_{i\tau}$, $\bar{W}_{ij\tau}$, $\bar{I}_{i\tau}$ and $\bar{B}_{i\tau}$ for all item $i$ and period $\tau$ in $\{t, ...T\}$ to represent the quantities, consumption, inventory, and backlog in the average scenario, and the following constraints are added to each subproblem:

$$\mathcal{F}_{t+1} \geq \sum_{\tau \in t...T} \left( \sum_{i \in \mathcal{I}} v_i \bar{Q}_{i\tau} + \sum_{i \in \mathcal{I}_c} h_i \bar{I}_{i\tau} + \sum_{i \in \mathcal{I}_e} \left( b_i \bar{B}_{i\tau} + \bar{I}_{i\tau} \right) + \sum_{i \in \mathcal{I}} \sum_{j \in \mathcal{I}} a_{ij} \bar{W}_{ij\tau} \right) \tag{44}$$

$$\widehat{Q}^{\xi}_{i\tau-L_i} + \widehat{I}^{\xi}_{i\tau-1} - \widehat{B}^{\xi}_{i\tau-1} - \bar{D}_{i\tau} - \bar{I}_{i\tau} + \bar{B}_{i\tau} = 0 \qquad i \in \mathcal{I}_e,\ \tau \in t+1 \dots T \tag{45}$$

$$\widehat{Q}^{\xi}_{i\tau-L_i} + \widehat{I}^{\xi}_{i\tau-1} - \sum_{j\in\mathcal{I}} \bar{W}_{ij\tau} - \bar{I}_{i\tau} = 0 \qquad i \in \mathcal{I}_c,\ \tau \in t+2 \dots T \tag{46}$$

$$\sum_{i\in\mathcal{A}_k} \bar{W}_{ij\tau} = R_{kj}\bar{Q}_{j\tau} \qquad k \in \mathcal{I}_c,\ j \in \mathcal{I},\ \tau \in t+1 \dots T \tag{47}$$

$$\bar{Q}_{i\tau} \leq M\widehat{Y}_{i\tau} \qquad i \in \mathcal{I},\ \tau \in t+1 \dots T \tag{48}$$

$$\sum_{i\in\mathcal{I}} \bar{Q}_{i\tau}K_{ir} \leq C_r \qquad r \in \mathcal{R},\ \tau \in t+1 \dots T \tag{49}$$

$$\bar{B}_{i\tau},\ \bar{I}_{i\tau},\ \bar{Q}_{i\tau} \geq 0 \qquad i \in \mathcal{I}_e,\ \tau \in t+1 \dots T \tag{50}$$

$$\bar{I}_{i\tau},\ \bar{Q}_{i\tau} \geq 0 \qquad i \in \mathcal{I}_c,\ \tau \in t+2 \dots T. \tag{51}$$

Constraints (44) link the lower bound with the expected cost-to-go. Constraint (45) and (46) compute the inventory and backlog for each item, where $\widehat{Q}^{\xi}_{i\tau-L_i}$, $\widehat{I}^{\xi}_{i\tau-1}$ and $\widehat{B}^{\xi}_{i\tau-1}$ correspond (depending on the value of $\tau$) either to the values found at an earlier stage, the decision variables of stage $t$, or the decisions of the average scenario. Finally, Constraints (47), (48), (49), (50), (51) correspond to consumption, production, capacity and domain constraints.

### 4.3.3 Multi-cut SDDP

In the multi-cut version of SDDP, the cost-to-go $\mathcal{F}^{\omega}_t$ is computed independently for each scenario $\omega$ of the next stage, and the expected cost-to-go is computed as

$$\mathcal{F}_t = \sum_{\omega\in\Omega_{t+1}} p_\omega \mathcal{F}^{\omega}_t.$$

In other words, the cuts are added separately for each scenario $\omega \in \Omega_{t+1}$. For instance, the cut added for scenario $\omega$ in stage $T-1$ is

$$\mathcal{F}^{\omega}_T \geq \sum_{i\in\mathcal{I}_e} \lambda^{l\omega}_i \left(-\widehat{Q}^{\xi}_{iT-L_i} - I^{\omega}_{iT-1} + B^{\omega}_{iT-1} + D^{\omega}_{iT}\right).$$

On the one hand, the multi-cut version leads to a better approximation of the cost-to-go functions. On the other hand, as the number of constraints increases, the multi-cut version can potentially slow down the solution process.

### 4.3.4 Advanced scenario sampling

The set of all possible demand scenarios is usually too large to solve the stochastic optimization problem. In this context, SDDP relies on a sample of scenarios, and the sampling method has a critical impact on the performance of SDDP (Parpas et al. 2015). For instance, the simple and efficient Crude Monte Carlo (CMC) (Metropolis and Ulam 1949) samples independently $n$ demand scenarios, and each scenario has a probability $1/n$. While the CMC approximation is on average equal to the expected cost, the variance of the approximation is $\sigma^2/n$, where $\sigma^2$ is the variance of the expected optimal cost. Advanced scenario sampling methods such as Randomized Quasi-Monte Carlo (RQMC) (Cranley and Patterson 1976) generate samples resulting in approximations with lower theoretical variances than CMC. In other words, RQMC leads to good approximations with fewer scenarios. As in Thevenin et al. (2020), to sample the demands with RQMC, we use the Lattice Builder tool (a software which implements multiple algorithms to build good rank-1 lattice rules) (L'Ecuyer and Munger 2016) to generate $n$ vectors $\boldsymbol{\alpha}$ (i.e. a lattice) evenly distributed in the cube $|\mathcal{I}_e|$. The demand vectors are then obtained by $\Omega = \{i\cdot\boldsymbol{\alpha}/n + \boldsymbol{\delta} \bmod 1 \quad \forall\, i \in 1 \dots n\}$, where $\boldsymbol{\delta}$ is a random value used to shift the lattice.

In this work, the scenarios $\Omega_t$ of the backward pass are sampled and fixed, whereas the forward pass scenarios $\Xi$ are re-sampled in each iteration from the original distribution.

# 5 Hybrid heuristics with SDDP

Despite the proposed improvements, the experiments (see Section 6) show that SDDP takes too much time to converge. However, the method spends most of the computation time on solving the first stage MILP. If the setup variables are fixed, the MILP becomes a linear program (LP), and SDDP converges in a reasonable amount of time. This section presents two heuristic strategies to compute the setups, namely a progressive hedging approach (Section 5.1), and a heuristic variant of SDDP (Section 5.2).

## 5.1 Progressive hedging

This section presents a method that determines the setups with a scenario tree approach. However, since planning over a long time horizon leads to large scenario trees, we consider progressive hedging (PH) to speed up the solution process.

PH (Rockafellar and Wets 1991) decomposes the original stochastic programming formulation into a set of deterministic problems, one per scenario. Similar to Lagrangian relaxation, PH relaxes the non-anticipativity constraints (7), but it penalizes their violation in the objective function. The non-anticipativity constraints (7) enforce the variables $Q^{\omega}_{it}$ (resp. $W^{\omega}_{ijt}$) to be equal for all scenarios $\omega$ in the set $NA(D^{\omega}_{1\dots t-1})$ of scenarios with identical demand up to period $t-1$. Also, the model (1)–(10) can be expressed with scenario-dependent setup variables ($Y^{\omega}_{it}$) if some additional non-anticipaticity constraints are included to force these variables to be equal for all scenarios. Once the non-anticipativity constraints are relaxed, the mathematical model (1)–(10) can be decomposed per scenario, but the resulting solution might not be implementable. To retrieve an implementable solution $\widetilde{S}$, the quantity $\widetilde{Q}^{\omega}_{it}$ and consumption $\widetilde{W}^{\omega}_{ijt}$ are averaged over the scenarios in $NA(D^{\omega}_{1\dots t-1})$, whereas the setups are averaged over all scenarios:

$$\widetilde{Q}^{\omega}_{it} = \frac{\sum_{\omega \in NA(D^{\omega}_{1\dots t-1})} p_{\omega} Q^{\omega}_{it}}{\sum_{\omega \in NA(D^{\omega}_{1\dots t-1})} p_{\omega}}$$

$$\widetilde{W}^{\omega}_{ijt} = \frac{\sum_{\omega \in NA(D^{\omega}_{1\dots t-1})} p_{\omega} W^{\omega}_{ijt}}{\sum_{\omega \in NA(D^{\omega}_{1\dots t-1})} p_{\omega}}$$

$$\widetilde{Y}_{it} = \sum_{\omega \in \Omega} p_{\omega} Y^{\omega}_{it}.$$

The following term is included in the objective function, to penalize the deviation from the implementable solution:

$$\sum_{x \in \mathcal{V}} \Lambda^{k}_{x}(x - \widetilde{x}) + \frac{\rho}{2}(x - \widetilde{x})^2,$$

where $V = \{ W^{\omega}_{ijt},\ Q^{\omega}_{it},\ Y^{\omega}_{it}\ \forall\ i, j \in \mathcal{I},\ t \in \mathcal{H},\ \omega \in \Omega\}$ is the set of decision variables subject to non-anticipativity constraints, $\widetilde{x}$ is the implementable decision associated with variable $x$, $\rho$ is a parameter, and $\Lambda^{k}_{x}$ is the Lagrangian multiplier in iteration $k$. This Lagrangian multiplier is updated for each variable $x \in V$ at the end of each iteration as follows:

$$\Lambda^{k}_{x} = \Lambda^{k-1}_{x} + \rho(x^{k} - \widetilde{x}^{k}),$$

where $x^{k}$ is the value of variable $x$ at iteration $k$. Note that $\rho$ is a sensitive parameter, since it impacts the quality of the solution and the computation time required for convergence. A large value of $\rho$ leads to fast convergence to a sub-optimal solution, whereas a small value leads to slow convergence.

Finally, PH stops when the setup variables have converged, that is:

$$\sum_{i} \sum_{t} \left( \widetilde{Y}_{it} - Y^{\omega}_{it} \right) \leq \epsilon,$$

where $\epsilon$ is set to 0.01 in this paper.

As explained earlier, we use PH to get the setup values, then SDDP optimizes the production quantities. Even though the hybrid PH SDDP relies on a scenario tree, it is convenient in a reactive context, because PH is only used to compute the setups (first-stage variables), and SDDP is run afterward. Consequently, the problem can be resolved dynamically in short computation time thank to the approximation of the cost-to-go.

### 5.2 Heuristic SDDP

Heuristic SDDP starts with initial setup values and iteratively performs the following steps: (1) run SDDP with fixed setups until the lower bound stabilizes (that is 10 iterations without improvement of LB); (2) solve the first stage problem (where the setups are not fixed) to find potentially better setup values.

The initial setups are computed with the two-stage approximation of model (1)–(12). In this two-stage approximation, the quantity and consumption variables do not depend on the scenario. In other words, the resulting model ignores the dynamic decision framework. However, the setup values found with the two-stage approximation are feasible (but sub-optimal) with respect to the original model (1)–(12). As the two-stage model does not need a scenario tree, a good approximation of the stochastic process requires only a limited number of scenarios. Consequently, the two-stage model can be solved in a reasonable amount of time.

## 6 Experiments

Section 6.1 describes the considered instances, and Sections 6.2 to 6.5 present computational experiments to evaluate the performance of SDDP and the proposed heuristic methods.

The algorithms are implemented in Python with CPLEX 12.9, and the tests are performed with an Intel Xeon Brodwell EP E5-2630v4 2.20GHz processor. For all methods, the stopping criterion is a time limit of $900|T|$ seconds.

The evaluation of the proposed SDDP policies is based on a simulation over a set of 5000 scenarios (which are drawn independently from the scenario sample $\Omega$ used for optimization). Each scenario represents a possible realization of the demand for the entire planning horizon, and these scenarios are sampled independently with Monte Carlo following the probability distributions of the demands. Note that for a given instance, all methods are evaluated with the same set of scenarios. As mentioned earlier, SDDP reacts quickly to new information, and this simulation is not time-consuming. In fact, the evaluation corresponds to a forward pass of SDDP on the given set of scenarios. This simulation returns an unbiased cost estimate of using each method $m$, which we call approximated upper bound and denote $\widehat{UB}(m)$. In the analysis of the results, we report the percentage gap ($GAP$) between the approximated upper bound of the considered methods:

$$GAP = 100\ \frac{\widehat{UB}(m) - \widehat{UB}^{\star}}{\widehat{UB}^{\star}}, \tag{52}$$

where $\widehat{UB}^{\star}$ is the best approximated upper bound among all methods tested in the considered experiment.

### 6.1 Instances generation

The experiments are performed with the well-known multi-echelon lot-sizing instances from Tempelmeier and Derstroff (1996). As these instances do not include lead times and demand distributions, these elements are generated similarly to Thevenin et al. (2020) as explained in the Online Supplement.

To study the performance of the methods under various conditions while maintaining a reasonable number of instances, we select a subset of instances from Tempelmeier and Derstroff (1996) with a $TBO$ of 1 or 3, a capacity utilization of 90% or 50%, and the two BOM given in Figure 2.

We generate instances with different values for the planning horizon ($T \in \{2, 4, 6, 8, 10\}$), the number of alternates ($a \in \{0, 2, 4, 6\}$), and the substitution costs ($a_{ij} \in \{0, 0.1, 1\}$). We set the default values for these parameters to $T = 4$, $a_{ij} = 0.1$, and $a = 4$, and we vary one factor at a time. This leads to a total of 80 instances.


Assembly BOM General BOM

**Figure 2: Considered BOM**

## 6.2 Performance of SDDP

This section evaluates the performance of SDDP with different scenario sample sizes $|\Omega_t| \in \{5, 10, 20, 50, 100\}$. To avoid convergence and memory problems, the experiments are performed with the setups fixed to the solution of the two-stage approximation of model (1)–(12) and with the single-cut version of SDDP.

Table 1 reports the average GAP obtained with each sample size. Table 1 shows that the sample size is a sensitive parameter. A too small number of scenarios leads to a bad approximation of the stochastic process, whereas a too large number prevents SDDP from converging. In fact, the sample size should depend on the planning horizon length. For 2 periods, 100 scenarios per stage lead to the best solution, whereas for 8 periods, 5 scenarios per stage lead to the best results.

**Table 1: Average GAP obtained with SDDP for different number of scenarios per stage**

| | $\|\Omega_t\|$ | | | | |
|---|---|---|---|---|---|
| $\|T\|$ | 5 | 10 | 20 | 50 | 100 |
| 2 | 0.24 | 0.09 | 0.03 | **0.00** | **0.00** |
| 4 | 1.18 | 0.19 | **0.01** | 0.21 | 0.23 |
| 6 | 0.85 | 0.20 | **0.00** | 0.60 | 0.72 |
| 8 | **0.08** | 0.11 | 0.19 | 0.81 | 2.05 |
| 10 | **0.05** | 0.06 | 0.25 | 1.25 | 1.89 |
| Average | 0.82 | 0.16 | **0.05** | 0.38 | 0.59 |

In the rest of the experiments, SDDP is run with 20 scenarios per decision stage since this scenario number gives the smallest average GAP.

## 6.3 SDDP enhancements

To evaluate the impact of the proposed SDDP enhancements, this section reports the performance of multiple variants of SDDP. We compare SDDP with all enhancements and the setup fixed with the two-stage heuristic, denoted by *Default*, with Default without the multi-cut, Default without the generation of strong cuts, Default without the average scenario lower-bound, Default without RQMC sampling, and Default without fixed setups.

Table 2 reports the average percentage gap $GAP^{UB}$ (resp., $GAP^{LB}$ ) between the upper bound (resp. lower) obtained with each variant of SDDP and the upper (resp. lower) bound obtained with

the Default SDDP algorithm. In addition, Table 2 gives the estimated optimality gap for each method $GAP^{Opt}$. We detail below the computation of $GAP^{UB}$, $GAP^{LB}$, $GAP^{Opt}$ for each method $m$.

$$GAP^{UB}(m) = 100\ \frac{\widehat{UB}(m) - \widehat{UB}(\text{Default})}{\widehat{UB}(\text{Default})}$$
$$GAP^{LB}(m) = 100\ \frac{LB(m) - LB(\text{Default})}{LB(\text{Default})}$$
$$GAP^{Opt} = 100\ \frac{\widehat{UB}(m) - LB(m)}{LB(m)}$$

Table 2 shows that the binary setup variables in the first stage subproblem significantly slow down the convergence of SDDP, since the optimality gap is 46.58% when the setups are not fixed. In addition, RQMC sampling has a significant impact since removing this component increases the upper bound by 0.62%. Similarly, the multi-cut version improves the convergence since removing this enhancement increases the upper bound by 0.19%. On the contrary, the strong cuts, and the average scenario lower-bound have a limited impact, and they decrease the upper bound by 0.02% and 0.10%, respectively.

**Table 2: Performance of each improvement of SDDP**

| Removed component | None | Multi-cut | Strong Cuts | Avg scenario | RQMC | Fixed setup |
|---|---|---|---|---|---|---|
| $GAP^{UB}$ | - | 0.19 | 0.02 | 0.10 | 0.62 | 25.31 |
| $GAP^{LB}$ | - | -0.09 | 0.01 | 0.01 | -2.02 | -5.73 |
| $GAP^{Opt}$ | 0.07 | 0.36 | 0.08 | 0.16 | 2.84 | 46.58 |

Figure 3 shows the evolution of the lower bound for different variants of SDDP on the instance with general BOM, $|\mathcal{H}| = 6$, $TBO = 1$, a utilization of 90%, 4 alternates, and an alternate cost of 0.1. Note that Default without multi-cut performs more than 2000 iterations within the time limit, but the last iterations are not showed for the sake of presentation. Figure 3 confirms that the fixed setups strategy and the multi-cut version of SDDP speeds up the convergence of SDDP, and the versions of SDDP with and without strong cuts behave similarly. In addition, SDDP with RQMC converges to a better lower bound than SDDP with MC. Finally, retaining the average scenario improves significantly the lower bound, but after a sufficient number of iterations, the variant of SDDP without this improvement reaches the same lower bound as the default version. In the rest of the experiments, SDDP includes all the improvements.

## 6.4 Comparison of the heuristic SDDP

This section compares the performance of different heuristics:

- The two stage heuristic presented in Thevenin et al. (2020) (denoted 2S);
- The version of SDDP with the setups fixed to the value of the two-stage approximation (denoted 2S-SDDP);
- The version of SDDP where the setups are fixed to the value of PH (denoted PH);
- The heuristic version of SDDP with multiple forward/backward for each possible setups vector (denoted HSDDP).

For each instance characteristic, Table 3 reports the average GAP of each method. The detailed results are available in the Online Supplement. The results show that HSDDP outperforms 2S-SDDP, 2S and PH with average GAPs of 0.30% versus 0.74%, 2.04%, and 20.22%, respectively.

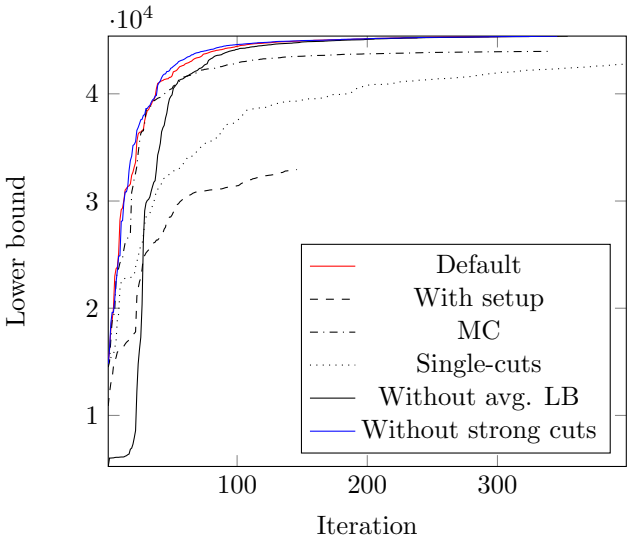


**Figure 3: Evolution of the lower-bound for different versions of SDDP**

**Table 3: GAPs of 2S, 2S-SDDP, PH, and HSDDP**

| | | 2S | 2S-SDDP | PH | HSDDP |
|---|---|---|---|---|---|
| $\|T\|$ | 2 | 0.01 | 0.00 | 8.43 | 0.13 |
| | 4 | 2.05 | 0.95 | 13.88 | 0.08 |
| | 6 | 2.23 | 0.96 | 21.16 | 0.45 |
| | 8 | 3.75 | 1.74 | 26.23 | 1.24 |
| | 10 | 3.99 | 1.73 | 27.32 | 0.39 |
| Structure | General | 2.33 | 1.02 | 18.63 | 0.23 |
| | Assembly | 2.22 | 1.07 | 25.30 | 0.22 |
| Utilization | 90% | 3.85 | 1.89 | 5.94 | 0.24 |
| | 50% | 0.70 | 0.20 | 37.99 | 0.21 |
| TBO | 1 | 1.83 | 0.67 | 22.30 | 0.31 |
| | 4 | 2.72 | 1.42 | 21.63 | 0.15 |
| Average | | 2.04 | 0.74 | 20.22 | 0.30 |

## 6.5 Comparison of the heuristic SDDP with optimal solutions

This section compares the results of the proposed SSDP approach with the optimal costs on instances with few scenarios. The optimal costs are computed with the model (1)–(12) and a symmetric scenario tree. Note that HSSDP is solved with the same set of scenarios as the MIP.

Table 4 provides the upper bound $\widehat{UB}$ obtained with HSDDP approximated based on a sample of 5000 scenarios, the percentage gap between the exact HSDDP upper bound (computed on all optimization scenarios) and the optimal solution, and the time at which the best solution was found. Note that some combinations of planning horizon and number of scenarios per stage lead to a huge number of scenarios. In such a case, it is impossible to compute the exact upper bound, and we indicate such instances using "-" in Table 4. For the MIP, Table 4 gives the actual CPU time spent to solve the instance, the optimality gap, and the number of variables and constraints. For some instances, the resulting mathematical model cannot be built since the size of the model was too large, and we indicate such instances using "-" in Table 4.

The results show that HSDDP performs well since the exact upper bound is only 1.2% (average of the values in the column "Gap exact UB" on the instances where the optimality gap can be obtained) larger on average than the optimal cost. The performance of the heuristic depends mainly on the

**Table 4: Comparison between HSDDP and the equivalent MIP**

| $\|\Omega_t\|$ | $\|T\|$ | $\|\Omega\|$ | $\widehat{UB}$ | HSDDP Gap exact UB (%) | HSDDP Time Best Sol. (s) | MIP CPUTime (s) | MIP Opt. Gap (%) | MIP Nr Variables | MIP Nr Constraints |
|---|---|---|---|---|---|---|---|---|---|
| 2 | 2 | $2^2$ | 22,410.55 | 0.00 | 4.64 | 0.02 | 0.00 | 93.00 | 143.13 |
| | 4 | $2^4$ | 39,617.15 | 0.09 | 497.54 | 0.35 | 0.00 | 375.67 | 538.50 |
| | 6 | $2^6$ | 58,298.64 | 2.45 | 3860.15 | 14.91 | 0.00 | 1510.00 | 2,127.00 |
| | 8 | $2^8$ | 77,003.57 | 3.39 | 6212.35 | 703.92 | 0.01 | 6278.00 | 8,633.00 |
| | 10 | $2^{10}$ | 96,383.99 | 3.40 | 9023.68 | 9041.68 | 3.24 | 25053.00 | 34,333.00 |
| 5 | 2 | $5^2$ | 22,202.27 | 0.00 | 41.75 | 0.07 | 0.00 | 149.00 | 249.00 |
| | 4 | $5^4$ | 40,568.14 | 0.57 | 1370.17 | 6.84 | 0.00 | 2560.67 | 4,007.50 |
| | 6 | $5^6$ | 59,773.26 | 1.09 | 4604.08 | 5400.02 | 6.60 | 64343.00 | 101,852.00 |
| | 8 | $5^8$ | 79,088.43 | - | 7522.59 | - | - | - | - |
| | 10 | $5^{10}$ | 93,844.87 | - | 9631.96 | - | - | - | - |
| 10 | 2 | $10^2$ | 22,657.23 | 0.00 | 103.95 | 0.11 | 0.00 | 222.00 | 395.00 |
| | 4 | $10^4$ | 40,797.96 | 0.97 | 1792.03 | 212.05 | 0.00 | 17944.33 | 31,704.83 |
| | 6 | $10^6$ | 59,153.93 | - | 5553.89 | - | - | - | - |
| | 8 | $10^8$ | 76,976.11 | - | 7851.30 | - | - | - | - |
| | 10 | $10^{10}$ | 95,905.33 | - | 10123.11 | - | - | - | - |

"-" indicates the instances that are too large to build the mathematical model since it consumes too much memory.

number of periods in the horizon, and not on the number of scenarios. Indeed, the gap remains 0 when the number of scenarios increases from 2 to 10 if $|T| = 2$, whereas the gap increases from 0 to 3.40 when the planning horizon increases from 2 to 10 periods with $|\Omega_t| = 2$. Finally, these experiments show that, contrarily to SDDP, the MIP approach is limited to a small number of scenarios since the MIP cannot be generated for 6 periods and 10 scenarios, and such a small number of scenarios does not give a good approximation of the stochastic demand.

## 6.6 The impact of component substitution

To analyze the impact of component substitution on the production plan, Table 5 reports the values of different key performance indices (KPIs), including total cost, rate of on-time delivery, inventory cost, backlog cost, lost sales cost, production cost, and component cost, for different numbers of alternates, and alternate costs. We compare the performance of the expected demand model, the two-stage model, and the heuristic SDDP. The three approaches are simulated in a static-dynamic decision framework. That is, the expected demand and two-stage models are re-solved in each iteration in a receding horizon fashion to update the production and consumption quantities based on the observed demands.

First, Table 5 shows that component substitution leads to lot consolidation. For instance, in the expected demand framework, the number of setups decreases from 28 to 19 when the number of possible alternates increases from 0 to 6. Second, component substitution leads to risk pooling. In the static-dynamic framework, the total inventory decreases by around 300 units when the number of possible alternates increases from 0 to 4, whereas the proportion of on-time delivery remains constant at 81%. In other words, pooling the risks allows maintaining the same service level with less inventory.

Table 5 shows that the level of inventory remains stable for 2-S when the number of substitutions increases, but the percentage of on-time delivery increases (from 75.3 to 75.9 with 2-S when the number of substitutable components increases from 0 to 6). The two-stage model does not plan to reassign the component's production to react to the observed demand. Nevertheless, when using the two-stage model in a static-dynamic decision framework, the components get re-assigned (thus the service level is improved when the number of possible substitutions increases). However, as these reassignments are not foreseen, the model orders the large number of components required to hedge against demand uncertainty in the static framework (to ensure the production of each end-item separately), and thus the inventory level remains large. In other words, to pool the uncertainty of multiple end-items, the planning must be created with a multi-stage optimization model.

**Table 5: Impact (in terms of various KPIs) of the number of substitutable components and of the substitution cost for the expected demand model, the two-stage model, and the heuristic SDDP in the static-dynamic decision framework**

| Method | $a$ | $a_{ij}$ | Total costs | On Time (%) | Nr Setups | Setup cost | Inventory cost | Backorder cost | Lost sales cost | Produc. cost | Consump. cost |
|---|---|---|---|---|---|---|---|---|---|---|---|
| HSDDP | 0 | 0.1 | 41,377.0 | 81.0 | 28.25 | 17,648.0 | 14,037.8 | 4,013.5 | 3,110.9 | 2,566.8 | 0.0 |
| | 2 | | 40,672.2 | 80.9 | 25 | 17,090.7 | 13,828.3 | 4,028.7 | 3,114.4 | 2,569.0 | 41.1 |
| | 4 | | 40,479.4 | 81.0 | 24 | 16,914.7 | 13,716.7 | 4,080.4 | 3,147.3 | 2,526.2 | 94.1 |
| | 6 | | 40,718.6 | 80.8 | 20.5 | 17,157.0 | 13,841.4 | 3,925.2 | 3,024.3 | 2,585.4 | 185.2 |
| | 4 | 0 | 40,624.6 | 80.9 | 20.25 | 17,139.5 | 13,772.8 | 4,040.1 | 3,093.5 | 2,578.6 | 0.0 |
| | | 0.1 | 40,479.4 | 81.0 | 24 | 16,914.7 | 13,716.7 | 4,080.4 | 3,147.3 | 2,526.2 | 94.1 |
| | | 1 | 41,193.5 | 81.1 | 26.5 | 17,571.6 | 14,106.4 | 3,860.9 | 2,963.0 | 2,581.5 | 110.1 |
| 2S | 0 | 0.1 | 42,000.4 | 75.4 | 27.75 | 17,606.7 | 12,347.6 | 4,796.1 | 4,759.8 | 2,490.1 | 0.0 |
| | 2 | | 41,249.3 | 75.5 | 24 | 17,047.0 | 12,330.9 | 4,784.1 | 4,557.1 | 2,499.3 | 30.8 |
| | 4 | | 40,804.5 | 75.7 | 23.75 | 16,887.0 | 12,224.0 | 4,762.3 | 4,390.0 | 2,487.3 | 53.8 |
| | 6 | | 41,198.2 | 75.9 | 20 | 17,134.5 | 12,447.9 | 4,553.2 | 4,443.6 | 2,510.6 | 108.4 |
| | 4 | 0 | 41,223.0 | 75.8 | 20.25 | 17,152.0 | 12,475.7 | 4,572.7 | 4,509.6 | 2,512.9 | 0.0 |
| | | 0.1 | 40,804.5 | 75.7 | 23.75 | 16,887.0 | 12,224.0 | 4,762.3 | 4,390.0 | 2,487.3 | 53.8 |
| | | 1 | 41,726.5 | 75.8 | 27.5 | 17,586.7 | 12,467.5 | 4,562.2 | 4,604.5 | 2,489.4 | 16.1 |
| Expected demand model | 0 | 0.1 | 68,982.3 | 52.2 | 28 | 13,502.5 | 6,933.4 | 6,395.9 | 40,637.6 | 1,513.0 | 0.0 |
| | 2 | | 66,650.9 | 52.6 | 24 | 12,787.5 | 6,997.0 | 6,304.2 | 39,016.6 | 1,525.6 | 20.0 |
| | 4 | | 65,875.2 | 53.0 | 23.25 | 12,763.0 | 7,032.4 | 6,207.5 | 38,325.5 | 1,516.2 | 30.6 |
| | 6 | | 65,046.0 | 53.1 | 19 | 12,730.0 | 7,040.8 | 6,170.9 | 37,442.7 | 1,549.8 | 111.8 |
| | 4 | 0 | 66,396.0 | 52.7 | 19.75 | 13,040.0 | 7,005.7 | 6,314.6 | 38,505.1 | 1,530.6 | 0.0 |
| | | 0.1 | 65,875.2 | 53.0 | 23.25 | 12,763.0 | 7,032.4 | 6,207.5 | 38,325.5 | 1,516.2 | 30.6 |
| | | 1 | 67,219.7 | 52.5 | 27.5 | 13,322.5 | 7,007.0 | 6,357.7 | 38,973.2 | 1,522.6 | 36.6 |

# 7 Conclusion

This paper investigates the performance of the SDDP algorithm for the Capacitated Multi-Echelon Lot-Sizing Problem (CMLSP) with stochastic demand and component substitution. The experiments show that SDDP performs well when the setups are fixed. Consequently, two strategies are proposed to compute the setups, namely progressive hedging, and a heuristic version of SDDP. The experiments show that the heuristic SDDP outperforms progressive hedging and the two-stage heuristic proposed in Thevenin et al. (2020). The proposed method can solve CMLSP with static-dynamic demand uncertainty for medium size planning horizons. In addition, contrarily to the scenario tree approach, once good approximations of the cost-to-go are built, SDDP can react quickly when new information is available.

As component substitution has the effect of aggregating the demand for components, it pools the risks associated with the uncertain demand, and it reduces the setups. Despite its practical relevance, only a few papers exist on lot-sizing with component substitution under stochastic demand. Future research should investigate stochastic optimization for the dynamic-dynamic decision framework, as well as uncertainty in various parameters including lead times, yields, and production capacities.

# References

Aloulou MA, Dolgui A, Kovalyov MY (2014) A bibliography of non-deterministic lot-sizing models. International Journal of Production Research 52(8):2293–2310.

Balakrishnan A, Geunes J (2000) Requirements planning with substitutions: exploiting bill-of-materials flexibility in production planning. Manufacturing & Service Operations Management 2(2):166–185.

Begnaud J, Benjaafar S, Miller LA (2009) The multi-level lot sizing problem with flexible production sequences. IIE Transactions 41(8):702–715.

Birge JR, Louveaux F (2011) Introduction to Stochastic Programming, Springer, New York.

Bookbinder JH, Tan JY (1988) Strategies for the probabilistic lot-sizing problem with service-level constraints. Management Science 34(9):1096–1108.

Brahimi N, Absi N, Dauzère-Pérès S, Nordli A (2017) Single-item dynamic lot-sizing problems: An updated survey. European Journal of Operational Research 263(3):838–863.

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

Buschkühland L, Sahling F, Helber S, Tempelmeier H (2010) Dynamic capacitated lot-sizing problems: a classification and review of solution approaches. OR Spectrum 32(2):231–261, ISSN 1436-6304.

Crainic TG, Rei W, Hewitt M, Maggioni F (2016) Partial Benders decomposition strategies for two-stage stochastic integer programs. Tech. Rep. CIRRELT–2016–37, Université de Montréal, Canada.

Cranley R, Patterson TNL (1976) Randomization of number theoretic methods for multiple integration. SIAM Journal on Numerical Analysis 13(6):904–914.

Dowson O, Philpott A, Mason A, Downward A (2019) A multi-stage stochastic optimization model of a pastoral dairy farm. European Journal of Operational Research 274(3):1077–1089.

Erlenkotter D (1978) A dual-based procedure for uncapacitated facility location. Operations Research 26(6):992–1009.

Gallego G, Katircioglu K, Ramachandran B (2006) Semiconductor inventory management with multiple grade parts and downgrading. Production Planning & Control 17(7):689–700.

Geunes J (2003) Solving large-scale requirements planning problems with component substitution options. Computers & Industrial Engineering 44(3):475–491.

Grubbström RW, Wang Z (2003) A stochastic model of multi-level/multi-stage capacity-constrained production–inventory systems. International Journal of Production Economics 81:483–494.

Han G, Dong M, Liu S (2014) Yield and allocation management in a continuous make-to-stock system with demand upgrade substitution. International Journal of Production Economics 156:124–131.

Harris FW (1990) How many parts to make at once. Operations Research 38(6):947–950.

Haugen KK, Løkketangen A, Woodruff DL (2001) Progressive hedging as a meta-heuristic applied to stochastic lot-sizing. European Journal of Operational Research 132(1):116–122.

Hsu A, Bassok Y (1999) Random yield and random demand in a production system with downward substitution. Operations Research 47(2):277–290.

Huang D, Zhou H, Zhao QH (2011) A competitive multiple-product newsboy problem with partial product substitution. Omega 39(3):302–312.

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

Karimi B, Ghomi SF, Wilson J (2003) The capacitated lot sizing problem: a review of models and algorithms. Omega 31(5):365–378.

Lang JC, Domschke W (2010) Efficient reformulations for dynamic lot-sizing problems with product substitution. OR Spectrum 32(2):263–291.

L'Ecuyer P, Munger D (2016) Algorithm 958: Lattice builder: A general software tool for constructing rank-1 lattice rules. ACM Transactions on Mathematical Software (TOMS) 42(2):15.

Lohmann T, Hering AS, Rebennack S (2016) Spatio-temporal hydro forecasting of multireservoir inflows for hydro-thermal scheduling. European Journal of Operational Research 255(1):243–258.

Magnanti TL, Wong RT (1981) Accelerating benders decomposition: Algorithmic enhancement and model selection criteria. Operations Research 29(3):464–484.

Metropolis N, Ulam S (1949) The Monte Carlo method. Journal of the American Statistical Association 44(247):335–341.

Pacaud F, Carpentier P, Chancelier JP, De Lara M (2018) Stochastic optimal control of a domestic microgrid equipped with solar panel and battery. ArXiv preprint arXiv:1801.06479.

Papadakos N (2008) Practical enhancements to the Magnanti–Wong method. Operations Research Letters 36(4):444–449.

Parpas P, Ustun B, Webster M, Tran QK (2015) Importance sampling in stochastic programming: A Markov chain Monte Carlo approach. INFORMS Journal on Computing 27(2):358–377.

Pereira MV, Pinto LM (1991) Multi-stage stochastic optimization applied to energy planning. Mathematical Programming 52(1-3):359–375.

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

Quezada F, Gicquel C, Kedad-Sidhoum S, Vu DQ (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.

Rao US, Swaminathan JM, Zhang J (2004) Multi-product inventory planning with downward substitution, stochastic demand and setup costs. IIE Transactions 36(1):59–71.

Rockafellar RT, Wets RJB (1991) Scenarios and policy aggregation in optimization under uncertainty. Mathematics of Operations Research 16(1):119–147.

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

Shin H, Park S, Lee E, Benton W (2015) A classification of the literature on the planning of substitutable products. European Journal of Operational Research 246(3):686–699.

Soares MP, Street A, Valladão DM (2017) On the solution variability reduction of stochastic dual dynamic programming applied to energy planning. European Journal of Operational Research 258(2):743–760.

Sox CR (1997) Dynamic lot sizing with random demand and non-stationary costs. Operations Research Letters 20(4):155–164.

Tarim SA, Dogru MK, Özen U, Rossi R (2011) An efficient computational method for a stochastic dynamic lot-sizing problem under service-level constraints. European Journal of Operational Research 215(3):563–571.

Tarim SA, Kingsman BG (2006) Modelling and computing (Rn, Sn) policies for inventory systems with non-stationary stochastic demand. European Journal of Operational Research 174(1):581–599.

Tempelmeier H (2013) Stochastic lot sizing problems. Handbook of Stochastic Models and Analysis of Manufacturing System Operations, 313–344 (Springer).

Tempelmeier H, Derstroff M (1996) A Lagrangean-based heuristic for dynamic multilevel multiitem constrained lotsizing with setup times. Management Science 42(5):738–757.

Tempelmeier H, Hilger T (2015) Linear programming models for a stochastic dynamic capacitated lot sizing problem. Computers & Operations Research 59:119–125.

Thevenin S, Adulyasak Y, Cordeau JF (2020) Material requirements planning under demand uncertainty using stochastic optimization. Production and Operations Management URL http://dx.doi.org/10.1111/poms.13277.

Tunc H, Kilic OA, Tarim SA, Rossi R (2018) An extended mixed-integer programming formulation and dynamic cut generation approach for the stochastic lot-sizing problem. INFORMS Journal on Computing 30(3):492–506.

Valladão D, Silva T, Poggi M (2019) Time-consistent risk-constrained dynamic portfolio optimization with transactional costs and time-dependent returns. Annals of Operations Research 282(1-2):379–405.

Wei M, Qi M, Wu T, Zhang C (2019) Distance and matching-induced search algorithm for the multi-level lot-sizing problem with substitutable bill of materials. European Journal of Operational Research 277(2):521–541.

Wu T, Akartunalı K, Jans R, Liang Z (2017) Progressive selection method for the coupled lot-sizing and cutting-stock problem. INFORMS Journal on Computing 29(3):523–543.

Yaman H (2009) Polyhedral analysis for the two-item uncapacitated lot-sizing problem with one-way substitution. Discrete Applied Mathematics 157(14):3133–3151.