Les Cahiers du GERAD ISSN: 0711–2440

Optimizing multi-item slow-moving inventory using constrained Markov decision processes and column generation

S. Thevenin, Y. Adulyasak, E. Prescott-Gagnon, T. Moisan

G–2024–03

January 2024

Citation suggérée : S. Thevenin, Y. Adulyasak, E. Prescott-Gagnon, T. Moisan (Janvier 2024). Optimizing multi-item slow-moving inventory using constrained Markov decision processes and column generation, Rapport technique, Les Cahiers du GERAD G– 2024–03, GERAD, HEC Montréal, Canada.

Avant de citer ce rapport technique, veuillez visiter notre site Web (https://www.gerad.ca/fr/papers/G-2024-03) afin de mettre à jour vos données de référence, s'il a été publié dans une revue scientifique.

Suggested citation: S. Thevenin, Y. Adulyasak, E. Prescott-Gagnon, T. Moisan (January 2024). Optimizing multi-item slow-moving inventory using constrained Markov decision processes and column generation, Technical report, Les Cahiers du GERAD G–2024–03, GERAD, HEC Montréal, Canada.

Before citing this technical report, please visit our website (https://www.gerad.ca/en/papers/G-2024-03) to update your reference data, if it has been published in a scientific journal.

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.

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.

Dépôt légal – Bibliothèque et Archives nationales du Québec, 2024
– Bibliothèque et Archives Canada, 2024

Legal deposit – Bibliothèque et Archives nationales du Québec, 2024
– Library and Archives Canada, 2024

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

Optimizing multi-item slow-moving inventory using constrained Markov decision processes and column generation

Simon Thevenin [a]

Yossiri Adulyasak [b]

Eric Prescott-Gagnon [c]

Thierry Moisan [c]

[a] *IMT Atlantique, LS2N-CNRS, La Chantrerie, 44300 Nantes, France*

[b] *Department of Logistics and Operations Management, HEC Montréal & GERAD, Montréal, (Qc), Canada, H3T 2A7*

[c] *ServiceNow, Montréal (Qc), Canada, H2S 3G9*

simon.thevenin@imt-atlantique.fr
yossiri.adulyasak@hec.ca
ericprescottgagnon@gmail.com
thierry.moisan@gmail.com

January 2024
Les Cahiers du GERAD
G–2024–03

Abstract : Inventory management for slow-moving items is challenging due to their high intermittence and lumpiness. Recent developments in machine learning and computational statistical techniques allow us to leverage complex distributions such as zero-inflated distributions to better characterize the demand functions of slow-moving items. Nevertheless, exploiting such outputs in the decision-making process remains a challenging task. We present an inventory optimization framework based on a coupled, constrained Markov decision process (CMDP) that is directly compatible with discrete demand functions. This approach can leverage complex discrete lead-time demand functions including empirical and zero-inflated distributions. The objective is to jointly determine inventory policies for multiple items under multiple target levels, which include common inventory measures such as stockout levels, fulfillment levels, and expected number of orders. To overcome the dimensionality issue, we employ a decomposition method based on a dual linear programming formulation of the CMDP and several computational enhancements. We propose a branch-and-price approach approach to solve the CMDP model exactly and a column generation heuristic. We provide computational comparisons with the approach in the literature as well as computational experiments using real-world data sets. The numerical results show we can solve CMDP efficiently, and its use in conjunction with empirical and zero-inflated negative binomial distributions outperforms benchmark and traditional approaches. The proposed framework provides practitioners with an efficient, flexible, and constructive tool to jointly manage the inventory of multiple items.

Keywords : Multi-item inventory, constrained MDP, column generation

Acknowledgements: The second author (Yossiri Adulyasak) gratefully acknowledges the support of Digital Research Alliance of Canada, the Natural Sciences and Engineering Research Council of Canada (RGPIN–2021–03264), and the Canada Research Chair in Supply Chain Analytics. Part of this work was performed at JDA Software, now Blue Yonder where Eric Prescott-Gagnon and Thierry Moisan were working during the time of this study.

1 Introduction

Inventory management is a crucial step in the supply chain planning process. In practice, inventory is typically managed and controlled simultaneously for a large number of items by taking into account several operational and service constraints imposed across these items. Even though a number of inventory management approaches have been developed (Axsäter 2015), a relatively small portion of the literature is devoted to stochastic multi-item inventory planning. In addition, manufacturing and retail companies regularly face a challenge in managing inventory of slow-moving items which typically comprise a large number of products and can potentially account for up to 40% of the entire inventory (McKinsey & Company 2019). Traditional inventory management approaches, which typically rely on common demand distribution functions such as the Gaussian or Poisson distribution, cannot be effectively applied in this context due to the irregular demand patterns exhibited by such items, which include high intermittence and lumpiness. These common assumptions, and the emphasis on rigorous analytical results, often made solution frameworks impractically relevant and difficult to use by practitioners (Tiwari and Gavirneni 2007). Even though there are studies that propose analytical solutions that do not rely on the structure of the demand distribution for a single-item problem (e.g., Scarf (1960), Veinott (1966), Zipkin (2008)), it is not straightforward to extend such analytical frameworks to a stochastic multi-item inventory problem where target level constraints can be imposed on a group of items. In addition, these frameworks do not generally allow for the incorporation of general and complex cost functions, such as batch costs or step-size inventory costs, which may differ from those utilized in the aforementioned studies.

In reality, the inventory planner must simultaneously manage a number of products wherein inventory targets or service levels can be imposed on a group of products as a means to achieve inventory management objectives of the businesses and operations (Kelle 1989, Thonemann et al. 2002, Akçay et al. 2016). These items may have different demand characteristics, including challenging ones such as highly intermittent items, slow movers, or items with high demand variation. Even if the planner creates or selects a proper demand distribution function for each product, finding an optimal set of policies across multiple products that minimizes the overall inventory management cost while respecting global targets remains a complex challenge.

This research aims to fill the gap between the researchers and practitioners by proposing an inventory optimization framework that tackles real-world inventory management issues while using techniques that are flexible and can be understood by practitioners. To this end, we propose a stochastic optimization framework based on a constrained Markov decision process for multi-item inventory management with group service levels. The resulting approach is generic enough to handle a range of inventory situations, and it can accommodate various discrete distribution functions to describe uncertain demand, especially for slow-moving items. This aspect is important because there is a practical need for an approach that can handle the empirical distribution (Zhang et al. 2014). One way to resolve this issue is to rely on techniques that do not necessitate specific forms of parametric demand estimates (e.g., Ehrhardt (1984), Federgruen and Zipkin (1984), Graves and Willems (2000), Downs et al. (2001), Chu et al. (2005), Wang (2011)).

The contribution of this work is fourfold. **First,** we present an inventory model to jointly manage multiple items under uncertain lead-time demand, where the items must respect targets or resource constraints imposed on inventory states across multiple items. We model the problem as a coupled, constrained Markov decision process (CMDP) using a mixed-integer programming model (MIP) where each MDP represents a single-item inventory problem. With the MDP representation, we can directly account for general discrete lead-time demand functions in the transition function. This model can also incorporate different cost components such as step-size batch orders, inventory cost, backlogging cost, or lost sale cost. **Second,** We present several practical considerations to reduce the dimensionality of the problem. To solve this problem efficiently, we apply a Dantzig-Wolfe decomposition (Dantzig and Wolfe 1960, Desaulniers et al. 2006) technique to the original coupled CMDP. We decompose the problem into a master problem and multiple sub-problems, where each sub-problem represents a single-item inventory process. A branch-and-price algorithm that incorporates a column generation procedure in a branch-and-bound procedure is developed to solve this reformulation exactly for small- to medium-size instances. To solve large-scale instances of the problem, we consider a demand aggregation approach to reduce the number of states. In addition, we propose a column generation heuristic that incorporates a local search that relies on Monte Carlo sampling to deal with the explosion of the action-state space. The resulting approach can efficiently solve large-scale instances encountered in practice. **Third**, to ensure that the policy can be easily implemented by practitioners, we provide an algorithm to transform the inventory policy from the MDP model in the form of an (s, S) inventory management rule. We provide a pseudo-polynomial time algorithm to generate an optimal (s, S) policy in the subproblem of Dantzig-Wolfe reformulation. The resulting approach directly provides an optimal (s, S) policy for each item while ensuring that all the policies from all the items in the group collectively respect global targets and constraints. **Fourth**, to numerically validate the performance and effectiveness of the proposed CMDP-based approaches, we perform the computational experiments on datasets from the literature with a large number of products (e.g., up to 1,000 items). To this end, we employ different discrete demand functions including empirical and zero-inflated demand distribution functions (Zuur et al. 2009) and compare them with standard inventory management policies which are based on Gaussian and Poisson distributions. Cross-validation (out-of-sample testing) is performed to compare the performance and quality of the inventory decisions produced by this approach versus other approaches found in the literature.

The rest of this paper is organized as follows. Section 2 discusses a literature review on multi-item inventory problems. Section 3 presents the CMDP formulations multi-item inventory optimization problems, as well as the dual reformulations; the practical enhancements and dimensionality reduction approaches are described in Section 4; the computational experiments are presented in Section 5; finally, the conclusion is provided in Section 6.

2 Literature review

A majority of multi-item stochastic inventory optimization focuses on the case of base stock policy, where unit costs associated with inventory, order, and shortage can be incorporated. Table 1 summarizes the relevant literature in this research area. A stochastic multi-item inventory model, with warehouse capacity, was first considered in Veinott Jr (1965) for the problem with a periodic review system with dynamic demand under a base stock policy where the action space (order quantities) across products can be bounded. In this framework, only the unit holding cost, unit ordering cost, and unit shortage can be considered. The authors established the conditions for which the base stock policy is optimal when there is zero lead time (delivery lag) or positive, but single, delivery lead time across all items. Ignall and Veinott Jr (1969) extended the results of Veinott Jr (1965) and showed the conditions where a myopic policy is optimal. Beyer et al. (2001) builds upon the framework of Ignall and Veinott Jr (1969) and uses dynamic programming to show that when demands are independent and the cost functions are separable, the modified base stock policy is optimal. Choi et al. (2005) considered the same problem as in Ignall and Veinott Jr (1969) except for the case of unequal replenishment intervals (zero lead times) and proposed a heuristic to solve it. In addition to the fact that the papers above deal with a base stock policy and single warehouse constraint, they typically assume that the lead time is zero or that the lead time for all the items is equal. For the case with delivery lead time, Downs et al. (2001) considered the multi-item with stationary discrete demand estimates, base stock policy, and lost sales under multiple linear resource constraints, and proposed a linear programming (LP) model to solve the problem. More recently, Akçay et al. (2016) proposed an optimization framework to deal with the multi-item inventory system to determine the base stock policies that satisfy the joint service level imposed on the joint demand distribution function whereas the objective of this problem is to minimize the inventory investment cost.

In the context of production-inventory systems where the production resource is limited, Federgruen and Zipkin (1986) considered a single-item problem with an infinite horizon, with zero lead time, where the objective was to minimize average cost over the long run. DeCroix and Arreola-Risa (1998) extended this framework to multiple items with single source production capacity and proposed a heuristic to solve the problem. Ketzenberg et al. (2006) dealt with a similar problem, as in DeCroix and Arreola-Risa (1998), but with seasonal demand and then formulated the problem using dynamic programming. The authors proposed a heuristic to solve it and provided extensive computational experiments. For the case when a fixed charge can be taken into account, Johnson (1967) considered a multi-item inventory problem with an infinite horizon where a fixed ordering charge can be incorporated. This study, however, does not consider resource constraints or capacity.

Table 1: Summary of relevant literature on multi-item inventory models. The cost parameters include unit ordering cost (o), unit holding cost (h), unit stockout cost (s), and fixed ordering cost (O).

	Multi -item	Generic discrete demand distr.	Target level(s)	Cost parameters	Supply lead time	Inventory Policy	Capacity (hard) const(s).
Veinott Jr (1965)	Yes	No	No	o, h, s	Yes	Base stock	Yes
Johnson (1967)	Yes	No	No	O, o, h, s	Yes	(σ, S) & (s, S)	No
Ignall and Veinott Jr (1969)	Yes	No	No	o, h, s	Yes	Base stock	Yes
Federgruen and Zipkin (1986)	No	Yes	No	o, h, s	No	Base stock	Yes
DeCroix and Arreola-Risa (1998)	Yes	No	No	o, h, s	No	Base stock	Yes
Beyer et al. (2001)	Yes	No	No	o, h, s	No	Base stock	Yes
Downs et al. (2001)	Yes	Yes	No	o, h, s	Yes	Base stock	Yes
Choi et al. (2005)	Yes	No	No	o, h, b	No	Base stock	Yes
Ketzenberg et al. (2006)	No	No	No	o, h, s	No	Base stock	Yes
Akçay et al. (2016)	Yes	Yes	Single[#]	h	Yes	Base stock	No
Our paper	Yes	Yes	Multiple	General	Yes	MDP*	Yes

[#] The target level considered in Akçay et al. (2016) is imposed on the joint distribution of multivariate demand functions.
* The solution is based on the Markov decision process (MDP) and thus provides a state-solution mapping function that can be transformed into an inventory policy.

Our work differs from these papers as our focus is on a more general case of inventory policies (rather than a base stock policy) where discrete demand functions and a general cost function can be incorporated in conjunction with target constraints across multiple products. Due to the fact that the resulting CMDP model is complex and cannot be solved analytically, we employ a mathematical decomposition approach, i.e., branch-and-cut and column generation heuristic, to tackle this challenging inventory optimization problem.

Even though unconstrained MDPs can be used to model single-item production and inventory problems (e.g., Yin et al. (2002), Chang et al. (2013), Cheng and Sethi (1999)) , these models often suffer from the curse of dimensionality when the state and action space are large. In addition and more importantly, constrained MDP models (Altman 1999) are usually intractable due to the fact that specialized and highly efficient algorithms originally developed for an unconstrained MDP cannot be directly applied (Boutilier and Lu 2016).

In this work, we model a multi-item inventory model as a CMDP where the inventory decisions of each item are modeled using an undiscounted infinite-horizon MDP. In other words, the constraints are weakly coupled across multiple items. Each constraint can be imposed on either all the items (global constraints), on a subset of items in the group, or even on an individual item level (note that the latter case is the most common case in the inventory management literature (Axsäter 2015)). The main contributions of this paper are fourfold: First, we introduce a mixed-integer programming (MIP) formulation based on the CMDP representation for the multi-item inventory planning problem with multiple target levels; second, since this practical problem is highly complex and the solution of the original CMDP does not necessarily yield a deterministic (or implementable) policy, we employ a mathematical decomposition technique to guarantee such implementable policy and to efficiently solve the problem; third, we present several computational enhancements and dimensionality reduction approaches to improve its scalability; and finally, fourth, we perform extensive computational experiments to demonstrate the performance of the proposed approaches, as well as computational comparisons with the approaches presented in the literature and used in practice using a cross-validation process.

The computational implications are also discussed. In addition, we discuss how the framework can be extended when constraints must be imposed directly on the maximum inventory capacity.

3 CMDP-based inventory model and formulation

This section successively presents the considered inventory model, the formulation as a set of coupled MDPs, and the decomposition for resolution with column generation

3.1 Problem description

In this context, an inventory planner of a firm seeks to determine an inventory policy for each product in the product (item) set K. To achieve inventory objectives, one or multiple inventory targets or expected resource constraints must be respected. Each target can be either imposed on an individual item, a subset of items, or all the items in the set K. The firm must account for uncertain customer lead-time demand (i.e., customer demand during supplier lead time). In this work, our specific focus is on the uncertain lead-time demand which is described by a general discrete demand distribution. As in the case of most literature in inventory management, the lead-time demand is assumed to be stationary and orders do not cross in time Zipkin (1986). As a result, the inventory policy may be designed based on the lead-time demand distribution. In this study, we do not make any assumption on the specific form of parametric distribution, and the lead-time demand follows any distribution with discrete support. More precisely, such a distribution gives the probability $P^k(\tilde{d} = d)$ to observe a demand vector $d \in \tilde{D}^k$ during the lead times of each of item k, and $\sum_{d\in\tilde{D}}^{k} P^k(\tilde{d} = d) = 1$. The objective is to design a multi-item control policy. We do not make any assumption on the form of the inventory control policy. More specifically, the resulting policy can be implemented using a look-up policy that gives the quantity q_i^k to order for item k when the inventory position is i. Nevertheless, the methodology can be adapted to restrict the search to specific policies such as a min-max policy (i.e., an order is passed to replenish the stock of item k to level S^k every time the stock level goes below a reorder point s^k), which is usually adopted in practice due to its simplicity. As the lead-time demand follows a discrete distribution, the set I^k of possible inventory positions for item k is also discrete. In each period, the inventory policy determines the quantity to order, and these quantities are integers.

The inventory policy is designed to minimize the long-run expected costs, and in this work, we consider a generic cost function that is a function of the inventory position i^k and order quantity q_i^k associated with each item k. This long-run expected cost is given by:

$$\min \sum_{k\in K} \sum_{i\in I^k} \sum_{q\in Q^k(i)} c_{iq}^k P^k(i,q) \tag{1}$$

where $P^k(i,q)$ is the probability to have an inventory position i for item k and to order a quantity q, and c_{iq}^k gives the costs associated with an inventory position i and order quantity q for item k. Note that $P^k(i,q)$ depends on the selected inventory policy. This cost structure is generic enough to accommodate the cost components commonly encountered in the inventory management literature. Table 2 gives the component to include in c_{iq}^k for the costs commonly encountered in the inventory management literature.

Note that the cost parameters account for the cost during the lead time for each item. For the inventory cost associated with inventory level i during the lead time, the value of c_{iq}^k can be computed as follows:

$$c_{iq}^k = \sum_{l\in\tilde{L}^k} P^k(\tilde{l} \geq l) \left(\sum_{x=0}^{i} P^k(\tilde{d}_l = i - x)\ \bar{c}_x^k \right), \tag{2}$$

where $P^k(\tilde{l} > l)$ is the probability that an actual lead time is greater than or equal to l, $\tilde{L}^k$ is the support of the lead-time distribution, and $P^k(\tilde{d}_l = d)$ is the probability that the demand is equal to d during a leas time of l periods.

Table 2: Examples of cost component

Description	component to add to c^k_{iq}
Holding cost h^k for each unit in stock per period	$\begin{cases} h^k\ i & \text{if } i >= 0 \\ 0 & \text{otherwise} \end{cases}$
Backlog cost b^k per unit delivered late	$\begin{cases} 0 & \text{if } i >= 0 \\ -b^k\ i & \text{otherwise} \end{cases}$
Stockout cost s^k	$\sum_{d^k=i+q}^{\tilde{D}^k} s^k\ (d^k - i - q)\ P(d^k)$
Fixed ordering costs O^k when the quantity to order is larger than 0	$\begin{cases} 0 & \text{if } q = 0 \\ O^k & \text{otherwise} \end{cases}$
Variable unit cost o^k per unit ordered	$o^k\ q$

Inventory policies of multiple items are linked by a target or a resource constraint. Such a resource constraint may represent a service level across the different items, a constraint on the space available for inventory, a constraint on the production capacity, ... In this work, we aim to satisfy a set of generic resource constraints M, and each constraint imposes a bound R^m on the expected consumption of the resource $m \in M$. Denote by K_m the set of items associated with constraint m and by phi^{km}_{iq} the long-run expected consumption of resource m for inventory quantity i and order quantity q of item k. The following target level (or expected resource consumption) constraints can be generally defined as:

$$\sum_{k\in K_m} \sum_{i\in I^k} \sum_{q\in Q^k(i)} \phi^{km}_{iq} P^k(i,q) \leq R^m \qquad \forall m \in M, \tag{3}$$

where ϕ^{km}_{iq} is the consumption of resource m with an inventory position i and order quantity q for item k, and $P^k(i,q)$ is the long run probability to have an inventory position i and to place an order of q units. $P^k(i,q)$ depends on the selected inventory policy. For example, a resource constraint to represent a service level $(1-\alpha)$ on the average probability of no stockout of all the items can be written as follows

$$\frac{1}{K} \sum_{k\in K} \sum_{i\in I^k} \sum_{q\in Q^k(i)} P^k(\tilde{d} \leq i) P^k(i,q) \geq (1-\alpha), \tag{4}$$

where $P^k(\tilde{d} \leq i)$ denotes the probability that the demand during the lead time for item k is lower than i.

These expected resource consumption constraints (3) can represent a variety of requirements associated with inventory states and decisions including common target levels used in practice. These constraints limit the average resource consumption which can be calculated for every combination of inventory level and ordering quantity. Table 3 provides examples of average resource consumption constraints that can be applied in an inventory planning environment. The parameter v^k denotes the unit value of the item k, and $\tilde{D}^k$ is the set of possible lead-time demand for item k. The negative values in coefficients and the right-hand side of the resource constraints represent a minimum bound on the expected consumption of the resource.

3.2 Formulation

We present below the formulation of multi-item inventory planning based on an infinite horizon Markov Decision Process (MDP). An MDP is defined by a set of states, a set of actions, transition matrices that describe the probability of moving from one state to another, and a reward matrix associating a reward to each pair of actions and states. We consider an MDP for each item, and these MDPs are linked by the resource constraint. More specifically, the tuple $\left\langle I^k, Q^k, \boldsymbol{P}^k, \boldsymbol{C}^k \right\rangle$ defines the MDP for each item k where $\boldsymbol{P}^k = (p^k(i|j,q))$ defines the transition function and $\boldsymbol{C}^k = (-c^k_{iq})$ defines the reward

Table 3: Notable examples of average resource consumption constraints

Description	ϕ_{iq}^{km}	R^m
Minimum no-stockout probability ($0 \leq \alpha \leq 1$)	$\phi_{iq}^{k\alpha} = -P^k(\tilde{d} \leq i)$	$-(1-\alpha)$
Minimum fulfillment level ($0 \leq \beta \leq 1$)	$\phi_{iq}^{k\beta} = -\frac{\sum_{d \in \tilde{D}^k} \min\{d,i\} \cdot P^k(\tilde{d}=d)}{\sum_{d \in \tilde{D}^k} d \cdot P^k(\tilde{d}=d)}$	$-(1-\beta)$
Maximum expected number of orders (N)	$\phi_{iq}^{kN} = \begin{cases} 1 & \forall q > 0 \\ 0 & \text{otherwise} \end{cases}$	N
Maximum expected inventory value (B)	$\phi_{iq}^{kB} = i \cdot v^k$	B

(expressed as a negative cost). A policy is a mapping of states (inventory level) of item k to probability distributions over actions (i.e., the probability of ordering q units for item k given that the inventory position is i. The policy is *deterministic* if the same action is chosen for a state and thus there is only one action (order quantity) with a probability of one for each state. In the case of *stochastic* policy, an action is randomized according to the probability distribution over the set of possible actions. The decision maker aims to minimize the total expected cost of inventory management while ensuring the actions across all the items (sub MDPs) respect global resource constraints.

Note that for the case of a single-item MDP without resource constraints, the model is aligned with the one presented in Yin et al. (2002) and Chang et al. (2013) where the states represent the inventory positions of the items; the actions are ordering quantities; and the transition matrices are defined by the lead-time demand probability and order action during a lead time. The single-item MDP variant can be modeled as a linear programming (LP) model (Bello and Riano 2006). We extend such an LP representation to account for the resource constraint across the different MDPs. More specifically, in this formulation, the inventory of each item is controlled by an undiscounted infinite-horizon MDP and the solution from the MDP provides a state-decision mapping function of the ordering decision q based on the inventory level i.

The decision variables of this MDP variant represent the probability associated with a state and action. More specifically, in the case of inventory planning, the value of the decision variable x_{iq}^k is the probability of carrying an inventory quantity i and ordering quantity q for item k at the decision epoch. The probability of moving the inventory position of item k from j to i, between subsequent stages, when applying a reorder action q (which will arrive after the lead time and brings the inventory quality to i) is defined by the transition probability $p^k(i|j,q)$. This transition probability is directly derived from the lead-time demand probability distribution and the inventory flow conservation, which can be calculated as follows:

$$p^k(i|j,q) = P^k(\tilde{d} = j + q - i) \tag{5}$$

The calculations of these input parameters are agnostic to the optimization framework presented here and thus this allows flexibility to account for more complex cases as long as one can determine a lead-time demand probability for the transition functions. For instance, in the case where the firm faces yield uncertainty, the transition matrix becomes:

$$p^k(i|j,q) = \sum_{\kappa \in Q^k(i)} P^k(\tilde{d} = j + \kappa - i) P^k(\tilde{q} = \kappa | q), \tag{6}$$

where $P^k(Q = \tilde{q}|q)$ denotes the probability that the quantity of the good quality product equals $\tilde{q}$ units when a quantity $q \geq \tilde{q}$ is ordered. Interested readers can refer to Appendix 1 as well as Yin et al. (2002) for supplementary details on numerical examples of the construction of the transition matrix from demand distribution for the MDP model.

The LP of the CMDP-based multi-item inventory model can be formulated as:

$$\min_{\mathbf{x}} \sum_{k \in K} \sum_{i \in I^k} \sum_{q \in Q^k(i)} c_{iq}^k x_{iq}^k \tag{7}$$

$$\sum_{q\in Q^k(i)} x^k_{iq} - \sum_{j\in I^k}\sum_{q\in Q^k(j)} p^k(i \mid j,q)x^k_{jq} = 0 \qquad \forall i\in I^k, \forall k\in K \tag{8}$$

$$\sum_{i\in I^k}\sum_{q\in Q^k(i)} x^k_{iq} = 1 \qquad \forall k\in K \tag{9}$$

$$\sum_{k\in K_m}\sum_{i\in I^k}\sum_{q\in Q^k(i)} \phi^{km}_{iq} x^k_{iq} \le R^m \qquad \forall m\in M \tag{10}$$

$$x^k_{iq} \ge 0 \qquad \forall i\in I^k, \forall q\in Q^k(i), \forall k\in K. \tag{11}$$

The objective function (7) minimizes the expected annual total cost associated with the inventory decision. Constraints (8) ensure the probability flow conservation between inventory states. These constraints map the state space based on the given transition function derived from demand distribution, inventory state, and replenishment decision. More specifically, the total probability flow associated with inventory state i is equal to the sum of all the probabilities associated with possible inventory state i and ordering quantity q which can result in inventory level i after the lead time. Constraints (9) ensure that the total probability flow of the MDP of each item sums up to 1. Constraints (10) are target service level and expected resource consumption constraints imposed on groups of items. Finally, constraints (11) ensure the non-negativity of the x_{iq} decision variables. We further note that the proposed framework can also be adapted when the MDP of each item corresponds to a different MDP that can be modeled as an LP using state visitation probability variables $\boldsymbol{x}$ (e.g., discounted infinite-horizon MDP). In such a case, the constraints associated with variables $\boldsymbol{x}$ must be modified accordingly to calculate their corresponding state visitation probability Puterman (2014).

Because of constraint (10), there is no guarantee that extreme points of this constrained MDP model correspond to deterministic policies (Derman and Veinott Jr 1972), which are important to ensure that the resulting policies are implementable. More specifically, a deterministic policy satisfies the condition $\max_{q\in Q^k(i)} x^k_{iq} = \sum_{q\in Q^k(i)} x^k_{iq}$. In other words, there is only a maximum of one optimal action q for each state i. To obtain a deterministic policy, the following constraints can be imposed:

$$v^k_{iq} \ge x^k_{iq} \qquad \forall i\in I^k, \forall q\in Q^k(i), \forall k\in K \tag{12}$$

$$\sum_{q\in Q^k(i)} v^k_{iq} = 1 \qquad \forall i\in I^k, \forall k\in K \tag{13}$$

$$v^k_{iq} \in \{0,1\} \qquad \forall i\in I^k, \forall q\in Q^k(i), \forall k\in K \tag{14}$$

where ν^k_{iq} is a binary variable stating whether ordering quantity q is selected for inventory position i of item k or not. As ν^k_{iq} is binary, the resulting solution yields a deterministic policy. Constraints (12) link the probability flow x^k_{iq} variables to the ν^k_{iq} policy variables, and constraints (12) state that only a single (deterministic) action can be taken at each state.

3.3 Dantzig-Wolfe reformulation

The model (7)–(11) requires a large number of variables including the binary variables which transform the linear programming model into a mixed-integer linear programming one. To ensure that a solution can be obtained in a tractable manner, we apply the Dantzig-Wolfe decomposition technique (Desaulniers et al. 2006) to solve this problem. One can observe that, without the resource constraint (10) across multiple items, we can decompose and solve the model (8), (9), (11) independently for each item $k \in K$. Denoted by Z^k a set of possible policies defined by the convex hull (8), (9), (11) for a given item k. For any given solution vector $\bar{\boldsymbol{x}}^{\boldsymbol{k}}_{\boldsymbol{\pi}}$ which defines an extreme point π^k of the polytope of this convex hull, the long-run expected cost c^k_π of policy π^k and the long-run expected consumption θ^{km}_π of resource m by policy π^k can be obtained with the given formulas:

$$c^k_\pi = \sum_{i\in I^k}\sum_{q\in Q(i)} c^k_{iq}\bar{x}^k_{iq,\pi}$$

$$\theta^{mk}_\pi = \sum_{i\in I^k}\sum_{q\in Q(i)} \phi^{km}_{iq}\bar{x}^k_{iq,\pi} \qquad \forall m\in M.$$

Note that, in the case where all the policies can be fully enumerated and included in Z^k, finding an optimal policy under target-level constraints can be done by solving the following master problem (MP):

$$\min_{\mathbf{y}} \sum_{k\in K}\sum_{\pi\in Z^k} c_\pi^k y_\pi^k \tag{15}$$

$$\sum_{k\in K_m}\sum_{\pi\in Z^k} \theta_\pi^{km} y_\pi^k \le R^m \qquad \forall m\in M \tag{16}$$

$$\sum_{\pi\in Z^k} y_\pi^k = 1 \qquad \forall k\in K \tag{17}$$

$$y_\pi^k \in \{0,1\} \qquad \forall k\in K, \forall \pi\in Z^k. \tag{18}$$

where y_π^k is the binary decision variable stating whether policy π^k is selected or not. The objective function (15) minimizes the long-run expected cost, constraints (16) are target level constraints and constraint (17) ensures that exactly one policy is selected.

3.4 Column generation and branch-and-price procedures

Enumerating the complete set of policies Z^k may not be practical and scalable. A column generation (CG) approach can be employed to iteratively generate a subset of the policy solutions. In this framework, at each iteration, a linear relaxation of the model (15)–(18), with a restricted set of policies, is solved to obtain dual variables. Then, the sub-problem, which consists of (19)–(22) with the modified objective of finding the policy with the least reduced cost, is solved to generate a new policy. More specifically, the reduced cost of the variable y_π^k, corresponding to a policy π^k, is given by the following formula:

$$\hat{c}_\pi^k = c_\pi^k - \sum_{m\in M} \lambda_m \theta_\pi^{km} - \lambda^{0k}$$

where λ_m are the dual variables of expected resource consumption constraints (16) and $\lambda^{0,k}$ is the dual variable of constraint (17). Therefore, a set of sub-problems SP^k, one for each item $k\in K$ are solved to generate new policies.

$$SP^k : \min_{\mathbf{x}} \sum_{i\in I^k}\sum_{q\in Q^k(i)} ((c_{iq}^k - \sum_{l\in L}\lambda_l \phi_{iq}^{kl}) x_{iq}^k) - \lambda^{0,k} \tag{19}$$

$$\sum_{q\in Q^k(i)} x_{iq}^k - \sum_{j\in I^k}\sum_{q\in Q^k(j)} p^k(i \mid j,q) x_{jq}^k = 0 \qquad \forall i\in I^k \tag{20}$$

$$\sum_{i\in I^k}\sum_{q\in Q^k(i)} x_{iq}^k = 1 \tag{21}$$

$$x_{iq}^k \ge 0 \qquad \forall i\in I^k, \forall q\in Q^k(i). \tag{22}$$

The model SP^k is in fact an LP model of the single-item MDP but with a modified objective (19) for item k where the dual variables (Lagrangian multipliers) $\boldsymbol{\lambda}$ are obtained by solving the relaxed MP. Indeed, one can solve these sub-problems in parallel to speed up the solution process especially when the number of items in the group is large.

It is possible that the master problem may not be feasible especially in the initial iterations when the number of variables is small. Therefore, we can solve a modified MP where slack variables to resource constraints so that such constraints are soft. The objective function of the modified MP is then to minimize the sum of the slack. This initial step of the CG procedure continues until the objective value is zero (no slacks) and the generated columns (variables) are used as initial variables in the CG procedure when solving the LP relaxation of the model (15)–(18).

In this work, we consider two versions of the CG approach, namely, a heuristic and an exact approach. The heuristic version solves the LP relaxation of the problem to generate the columns.

Once the columns are generated, the heuristic imposes the integrality constraint to get a feasible (but not necessarily) optimal solution. To obtain an implementable solution, the master problem with integrality constraints (18) is solved to determine a valid upper bound, which is an optimal set of policies based on the restricted set of policies generated so far. An optimality gap can be determined at each iteration by using the lower bound from (15). The process continues until the sub-problem does not yield a negative reduced cost and thus no further policies can be generated.

The exact approach embeds the CG approach in a branch-and-bound procedure. This procedure is known as branch-and-price (B&P). In B&P, the LP relaxation of each branch-and-bound node is solved using CG. If the solution of the linear program is not integer, then there is at least one state that has at least two actions (two variables) with a positive probability. In that case, we branch on such a state by creating two possible branch-and-bound nodes where the associated variable is set to zero in one branching node, and set to one in the other branching node. Note that to simplify the implementation, rather than set the variable to one, we set all variables associated with the state to 0.

More precisely, we start by normalizing the actions associated with each state. The normalized action $\hat{x}^k_{iq}$ is computed as follows:

$$\hat{x}^k_{iq} = \frac{x^k_{iq}}{\sum_{q \in Q^k(i)} x^k_{iq}}.$$

In a deterministic policy, $\hat{x}^k_{iq}$ takes value 0 or 1. We consider the value $k^\star$, $i^\star$, $q^\star$ corresponding to the value of $\hat{x}^{k^\star}_{i^\star q^\star}$ the closest to 0.5 (the most fractional variable), and we create two branches. In the first branch, we impose $x^{k^\star}_{i^\star q^\star} = 0$ when solving the sub-problem, and we set $y^k_\pi = 0$ in the master problem if $x^{k^\star}_{i^\star q^\star,\pi} > 0$. In the second branch, we set $x^{k^\star}_{i^\star q} = 0$ for all $q \neq q^\star$ in the sub-problems, and we remove the corresponding policies in the master problem. The solution of the linear relaxation of the problem in each node provides a lower bound, and we get an upper bound when the solution of the linear relaxation is an integer. We use this upper bound to prune the nodes whose lower bound is larger than the upper bound. The approach stops when there are no active branch-and-bound nodes, or when a time limit is reached.

4 Computational and practical enhancements

In this section, we discuss several enhancements to improve the scalability: a pseudo-polynomial time algorithm to solve the sub-problem when the policy is restricted to a min-max rule, a local search policy generation that can solve large-scale instances, and dimensionality reduction through state aggregation. Enhancements to deal with practical issues are also presented: an extension to the problem with capacity constraints on the maximum resource utilization, and the complex distributions (zero-inflated demand distributions) used for irregular demand in practice.

4.1 Pseudo-polynomial algorithm for (s, S) inventory policy generation

Inventory control systems in practice are typically designed to take standard inventory policy parameters as input. For instance, the (s, S) inventory policy is commonly used in real-life inventory fulfillment systems. Such (s, S) policies can be readily used by companies to control their inventory fulfillment process. In this section, we propose an algorithm to solve the sub-problem when the policy must take the form of an (s, S) inventory control rule. The rest of this section first explains how to build the Markov Decision Process associated with the post-decision state of a given (s^k, S^k) policy for item k, before presenting an efficient computation of the long-run expected cost of the resulting MDP. Formally, the ordering quantity $\pi_{(s^k,S^k)}(i)$ for item k associated with inventory level i based on min-max inventory policy (s^k, S^k) can be defined as follows:

$$\pi_{(s^k,S^k)}(i) = \begin{cases} 0 & \text{if } i > s^k \\ S^k - i & \text{otherwise} \end{cases} \tag{23}$$

In other words, the ordering quantity equals zero if the inventory level $i > s^k$, and equals $S^k - i$ otherwise.

To transform the min-max policy (s^k, S^k) into a solution vector $\boldsymbol{x}^k$ of the original CMDP model given in Section 3.2, we determine the probability $w^k_{j'}$ that item k is in a prior *post-decision* state j' (i.e., $j' = j + q^*$ where j is the prior state and q^* is the quantity decision made in the prior state). More specifically, w^k_j is the probability of having inventory position (inventory on-hand plus the in-transit order) j' right after a prior order q^* has been placed. These post-decision states are linked with the original variables as follows:

$$x^k_{iq} = \begin{cases} \sum_{j'=s^k}^{S^k} P^k(\tilde{d} = j' - i) w^k_{j'} & \text{if } q = \pi_{(s^k,S^k)}(i) \\ 0 & \text{otherwise,} \end{cases} \tag{24}$$

where $\widetilde{P}^k(i'|j)$ is the probability to move to state i' from post decision state j. The transition matrix $\widetilde{P}^k(i|j)$ is derived directly from the transition probability used in the inventory model. The policy gives the quantity q to order for each inventory level i, and an action-state that does not respect the policy has a probability of 0. As there is a single quantity per inventory level, Equation 24 computes the probability of reaching state i from the post-decision state j'.

Given a (s^k, S^k) policy, we can define a Markov process for the post-decision states. As the policy orders up to S^k when the inventory level is below s^k, there are no post-decision states for inventory levels below s^k due to the fact that an order must be placed in a prior state and this immediately brought the inventory position j' to S^k. The probability of $j' > S^k$ is also zero because the demand is non-negative and the probability is strictly positive. The transition probability $\widehat{P}^k(i'|j')$ from post-decision state j' to post-decision state i' can be computed as follows:

$$\widehat{P}^k(i'|j') = \sum_{i \in I^k | i + \pi_{(s^k,S^k)}(i) = i'} P^k(\tilde{d} = j' - i). \tag{25}$$

Proposition 1. *The probability w^k_j of being in post-decision state j can be computed as follows $w^k_j = K^k_j w^k_S$, where $K^k_S = 1$ and $K^k_j = \sum_{i \in j+1 \ldots S^k} \frac{\widehat{P}^k(j|i)}{1 - \widehat{P}^k(j|j)} K_i$, and $w^k_S = \frac{1}{\sum_{i \in \{s^k \ldots S^k\}} K^k_i}$.*

(Note that Proposition 1 is not valid when $\widehat{P}^k(j|j) = 1.0$ which corresponds to the case of zero demand with a probability of 1.0, but this case is not relevant in our context).

Proof. The (s^k, S^k) policy orders up to S^k when the inventory position is below s^k. Therefore, if the post-decision MDP is in an inventory position j, it can transition to an inventory position $k \leq j$ if the demand is positive and the order quantity is 0, or it transitions to inventory position S^k. In other words, as we assume the demand is positive, the post-decision MDP can only transition to an inventory position i strictly lower than S^k from an inventory position j larger than i ($\widehat{P}^k(i|j) = 0 \; \forall j < i < S^k$). Therefore, the long run average probabilities satisfy:

$$w^k_i = \sum_{j \in \{i, \ldots, S^k\}} \widehat{P}^k(i|j) w^k_j \qquad \forall i \in s^k, \ldots, S^k - 1$$

As a result, the probability w^k_i to be in post-decision state i can be expressed recursively as a function of w^k_S. For state $S^k - 1$,

$$w^k_{S-1} = \widehat{P}^k(S^k - 1 | S^k - 1) w^k_{S^k-1} + \widehat{P}^k(S^k - 1 | S^k) w^k_S. \tag{26}$$

Thus,

$$w^k_{S-1} = \frac{\widehat{P}^k(S^k - 1 | S^k)}{1 - \widehat{P}^k(S^k - 1 | S^k - 1)} w^k_S. \tag{27}$$

More generally, if states $j+1$ to S^k-1 are expressed as a function of w^k_S ($w^k_i = K^k_i w^k_S$ $\forall i \in \{j+1,\ldots,S^k-1\}$):

$$w^k_j = \sum_{i\in\{j,\ldots,S^k\}} \widehat{P}^k(j|i)w^k_i \tag{28}$$

$$= \widehat{P}^k(j|j)w^k_j + \sum_{i\in\{j+1,\ldots,S^k\}} \widehat{P}^k(j|i)K^k_i w^k_S \tag{29}$$

$$= \sum_{i\in\{j+1,\ldots,S^k\}} \frac{\widehat{P}^k(j|i)}{1-\widehat{P}^k(j|j)} K^k_i w^k_S \tag{30}$$

Finally, as $\sum_{i\in\{s^k,\ldots,S^k\}} w^k_i = 1$, $\sum_{i\in\{s^k,\ldots,S^k\}} K^k_j w^k_S = 1$, and

$$w^k_S = \frac{1}{\sum_{i\in\{s,\ldots,S\}} K^k_j}$$

□

Remark 1. The value of w^k_{j-s} is the same for all (s^k,S^k) policies with the same length (S^k-s^k).

Remark 2. The cost of an (s^k,S^k) policy can be directly computed based on the post-decision state probability.

$$\sum_{i\in I^k} c^k_{i\pi^k_{(s,S)}(i)} x^k_{i\pi_{(s,S)}(i)} = \sum_{i\in I^k} c^k_{i\pi_{(s,S)(i)}} \sum_{j=s^k}^{S^k} P^k(\tilde{d}=j-i)w^k_j$$

$$= \sum_{j=s^k}^{S^k} \left(\sum_{i\in I^k} c^k_{i\pi_{(s,S)(i)}} P^k(\tilde{d}=j-i) \right) w^k_j$$

Remark 3. In the special case where the costs do not depend on the quantity ($c^k_{iq} = c^k_i$), it is optimal to set $w^k_{j^\star} = 1$ for the state $j^\star$ that minimizes $\sum_{i\in I^k} c^k_i P^k(\tilde{d}=j^\star-i)$. The resulting policy is a base stock ($s^k = S^k$).

Algorithm 1 give the pseudo code to compute the optimal values of s^k and S^k based on proposition 1.

Algorithm 1 A pseudo-polynomial time algorithm to determine an optimal (s^k,S^k)

$K^k_1 = 1$
for $l \in 1\ldots|I^k|$ **do**
  Compute the value of $K^k_{|I^k|-l}$ for a police of length l

$$K^k_{|I^k|-l} = \sum_{i=|I^k|-l+1}^{|I^k|} \frac{\widehat{P}^k(|I^k|-l|i)}{1-\widehat{P}^k(|I^k|-l||I^k|-l)} K^k_i$$

  Compute the value of the value of w^k_S with $V^k_K = V^k_K + K^k_{S^k-l}$ and $w^k_S = \frac{1}{V^k_K}$
  for $s^k \in 0\ldots|I^k|-l$ **do**
    Compute the cost of the polity (s^k, s^k+l)

$$C = \sum_{j=s^k}^{s^k+l} \left(\sum_{i=1}^{|I^k|} c^k_{i\pi_{sS}(i)} \widetilde{P}^k(i|j) \right) K^k_j w^k_S$$

.
    If C is lower than the current best policy, record the policy (s^k, s^k+l).
  end for
end for

Remark 4. Algorithm 1 runs in pseudo-polynomial time $\mathcal{O}\left((\max_{k\in K}|I^k|)^3\right)$

4.2 Fast policy generation using local search

Solving the sub-problem (19)–(22) with a commercial solver for one slow-moving item can be relatively fast, but when we need to generate policies for thousands of items of for fast movers, we quickly reach the limits of memory and processing time. Similarly, Algorithm 1 is not practical for large values of $|I^k|$. To increase the scalability of the general approach, a local search heuristic is proposed to solve (19)–(22) within the column generation procedure.

The local search explores the space of (s^k, S^k) policies by changing the values of s^k and S^k parameters to quickly generate a set of good policies for item k. To ensure that the generated policies will potentially improve the global objective across multiple items, the policies generated through the local search procedure are evaluated using a Markov chain with the same reward and transition functions used in the MDP for SP^k (which makes use of the dual information obtained from the MP to guide the search) but the actions (replenishment quantities) are fixed based on the (s^k, S^k) policy determined during the local search procedure. To avoid scalability issues during the evaluation of a policy, it can also be evaluated using Monte Sampling where a large number of sample paths (1000 in our experiments) are generated and the policy is evaluated on these paths. The first periods in each path (the first 10% in our experiment) are initialized in to let the chain reach its equilibrium state. Then, we can generate and add multiple inventory policies to the master problem at each iteration by executing this local search procedure and storing a set of multiple policies that are generated during the local search. Appendix 2 in the online supplement provides the details of the procedure.

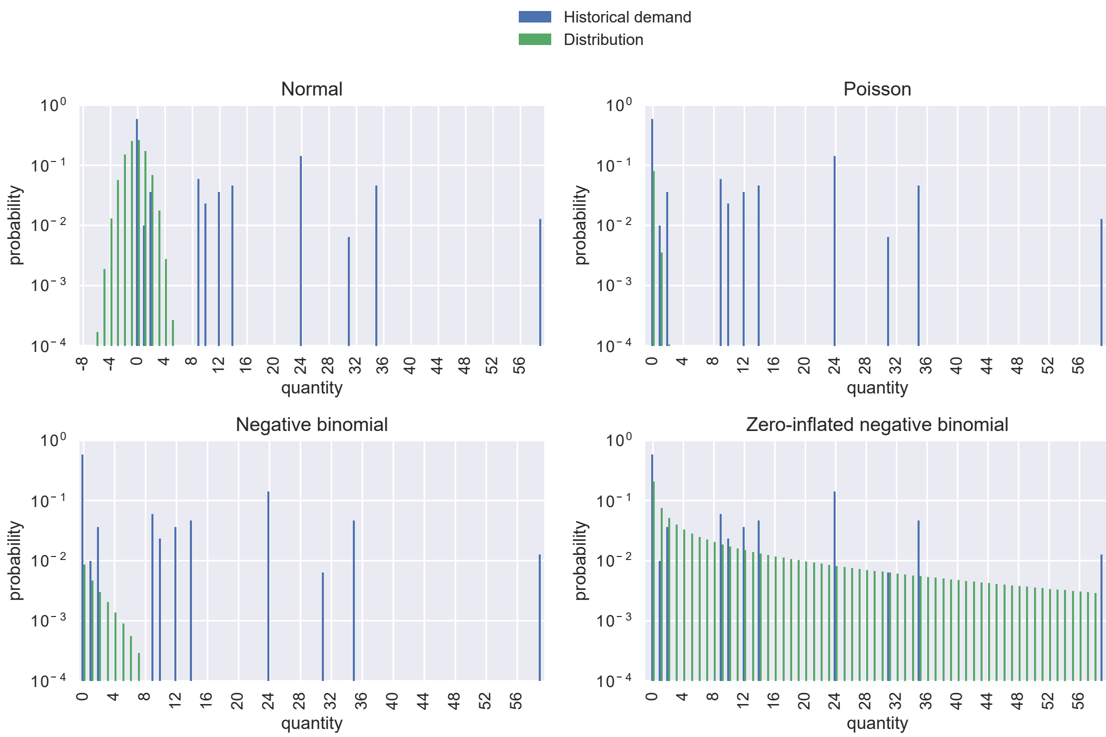
4.3 State aggregation using an aggregation factor

Large quantities of inventory states and orders can quickly make a state space too large to be handled efficiently. Instead of enumerating all possible states, the inventory quantities can be aggregated into buckets of a predetermined size. This can be achieved by creating a factored demand function, based on an aggregation factor, instead of explicitly considering the original demand values (Yin et al. 2002), whereas the subsequent process remains the same. For example, given an aggregation factor $\tau^k = 2$ for an item k, the original set demand values $\tilde{D}^k = \{0, 1, 2, ..., 10\}$ become $\tilde{D}^k = \{0\tau, 1\tau, 2\tau, ..., 5\tau\}$. This aggregation scheme through an aggregation factor can help improve solving time, memory usage, and solution quality significantly as we demonstrate in the experimental results section.

4.4 Demand distribution considerations for irregular items

Finding a distribution matching the demand history of irregular items such as slow movers can be difficult (Dolgui and Pashkevich 2008). Poisson distribution, which is commonly assumed, cannot adequately capture the demand patterns in the case when the intermittent demand is also very lumpy (i.e., the positive demand can be arbitrarily large). In such cases, an empirical distribution or zero-inflated distribution (e.g., Zuur et al. (2009), Chapados (2014)) can be effectively used to better represent the demand distribution. Figure 1 shows a typical slow-mover product demand probability versus four of the most frequently used demand distribution functions, namely: Gaussian, Poisson, negative binomial, and zero-inflated negative binomial. The Gaussian distribution misrepresents the demand distribution and clearly misses all demand quantities above 10. The Poisson distribution handles the probability of zero demand, but cannot capture high-demand quantities. The negative binomial distribution better captures high-demand quantities compared to the Poisson distribution, but it underestimates the high probability of zero demand and still underestimates the positive demand quantities. Finally, the zero-inflated negative binomial distribution spreads out the probability mass more appropriately on all possible demand quantities, while sufficiently capturing the zero demand probability. This shows the difficulty in matching such demand patterns with typical statistical distributions. Even though a more general form (i.e., zero-inflated distributions) can be used to better represent the demand for irregular items, traditional inventory management approaches that rely on specific functional forms cannot be directly applied whereas our CMDP-based model, which takes a generic discrete lead-time demand distribution function as inputs, can be efficiently leveraged.

Figure 1: The empirical distribution based on the historical demand of an item compared to four distributions fitted to its historical demand. The probability axis is in the log scale



As described above, selecting the right demand function estimate and inventory management technique for each product is already a very challenging task for an inventory planner. For the items with irregular demand that do not exhibit the demand distributions that resemble any of the standard distributions (in particular the Poisson and Gaussian distributions), the companies can choose to use an empirical distribution or zero-inflated distributions (Zuur et al. 2009) to describe the demand distributions for these items. The choice of empirical distribution comes from the fact that it is easily understood by decision-makers and planners. In terms of data management, however, it can be more practical to use parametric discrete distributions (which include complex ones such as zero-inflated distributions) since the firm needs to maintain only the parameters of the distribution rather than keeping all the values describing the empirical distribution (i.e., the probability associated with each possible demand value). In addition, parametric discrete distributions can also help reduce over-fitting caused by the empirical distributions generated from a limited amount of historical data (Long et al. 2023).

One particular zero-inflated distribution that we use in our experiments is the zero-inflated negative binomial distribution (ZINB) which has been effectively used to generate probabilistic demand functions for slow-moving demand (Chapados 2014). This distribution represents the demand probability in the following parametric form:

$$\tilde{d} \sim z\delta_0 + (1-z)\mathrm{NB}(r,p) \tag{31}$$

where z is the probability that the demand is zero and δ_0 represents unit probability mass at zero. $\mathrm{NB}(r,p)$ represents a negative binomial distribution with its corresponding parameters r and p. The distribution is, in fact, a mixture distribution where the probability of zero demand equals z and the probability of positive demand, which corresponds to $\mathrm{NB}(r,p)$, equals $(1-z)$. To determine the parameters z, r, and p for the ZINB, we use the approximate inference technique presented in Chapados (2014) that minimizes the log-likelihood of the approximation of parameters z, r and p from the observed demand data. Note that this demand function can be also approximated by other applicable algorithms such as the expectation propagation (EP) and Markov chain Monte Carlo (MCMC) (e.g., see Minka (2001)).

The lead-time demand distribution $P^k(\tilde{d} = x)$ may be computed based on the lead-time distribution $P^k(\tilde{l} = l)$ and the demand per period distribution $P^k(\tilde{d}_l = d)$, as follows:

$$P^k(\tilde{d} = x) = \sum_{l=1}^{L} P^k(\tilde{l} = l) P^k(\tilde{d}_l = x), \tag{32}$$

where $P^k(\tilde{d}_l = x)$ denotes the probability that the cumulative demand of l consecutive period is equal to x. $P^k(\tilde{d}_l = x)$ can be computed from the per-period demand distribution as follows:

$$P^k(\tilde{d}_l = x) = \sum_{x_1 \leq x_2 \leq \ldots \leq x_{l-1} \leq x} P(\tilde{d}_1 = x_1) P(\tilde{d}_1 = x_2 - x_1) P(\tilde{d}_1 = x_3 - x_2) \ldots P(\tilde{d}_1 = x - x_{l-1}), \tag{33}$$

where x_1, x_2, $\ldots$, x_{l-1} may be seen as the cumulative demand. However, such computation is computationally intensive, and the estimation of $P(d_l = x)$ with bootstrap (moving block sampling) is more efficient. More precisely, we randomly sample $10,000$ sets of l demand values from the historic demand, and the sum of the demand corresponds to a possible value of the demand during l periods. This sample is directly used as input to the probabilistic forecasting method.

This process is performed as a pre-processing step to create the input for the optimization model presented in this paper. Therefore, the computational performance of the approximation algorithm does not have an impact on the computing time of our CMDP model.

4.5 Extension of the model to restrict the maximum resource consumption

Constraints (3) restrict the expected resource utilization, but they can also deal indirectly with the capacity constraints on the maximum resource utilization (e.g., inventory or space capacity). One practical way to deal with these types of capacity constraints is to consider a resource constraint violation probability and use the constraint derived from Hoeffding's inequality (Hoeffding 1963) by adapting the approach presented in de Nijs et al. (2017).

Denote by X^{km} and U^{km} the resource utilization and the upper limit on the use resource m of item k, respectively. Let C^m denote the capacity for resource m and κ^m denote the upper bound on the probability of capacity constraint violation of resource m. ϕ_{iq}^{km} still represents the resource consumption when the inventory position i for item k and an order of quantity q is placed. For instance, to model a maximum inventory level $\phi_{iq}^{km} = iv^k$, where v^k is the volume of a unit of item k.

Proposition 2. *The constraint guaranteeing that the probability of violating the limit C^m will not exceed κ_m, i.e, $\mathbb{P}\left(\sum_{k\in K} X^{km} > C^m\right) \leq \kappa^m$ is given by:*

$$\sum_{k\in K} \sum_{i^k \in I^k} \sum_{q^k \in Q^k(i)} \phi_{iq}^{km} P^k(i, q)) \leq C^m - \sqrt{\frac{ln(\kappa^m) \cdot (\sum_{k\in K} (U^{km})^2)}{-2}} \tag{34}$$

Proof of proposition 1.

$$\mathbb{P}\left(\sum_{k\in K} X^{km} > C^m\right) = \mathbb{P}\left(\sum_{k\in K} X^{km} - \mathbb{E}(X^{km}) > C^m - \mathbb{E}(X^{km})\right) \tag{35}$$

with $\mathbb{E}(X^{km}) = \sum_{k\in K}\sum_{i^k\in I^k}\sum_{q^k\in Q^k(i)}\phi_{iq}^{km}P^k(i,q))$. Based on Hoeffding's inequality, the probability to exceed the capacity is bounded by:

$$\mathbb{P}\left(\sum_{k\in K} X^{km} > C^m\right) \leq exp(\frac{-2(C^m - \sum_{k\in K}\sum_{i^k\in I^k}\sum_{q^k\in Q^k(i)}\phi_{iq}^{km}P^k(i,q)))^2}{\sum_{k\in K}(U^{km})^2}). \tag{36}$$

Therefore, the long-run probability to exceed the capacity is lower than κ^l if

$$\kappa^m \geq exp(\frac{-2(C^m - \sum_{k\in K}\sum_{i^k\in I}\sum_{q^k\in Q^k(i)}\phi_{iq}^{km}P^k(i,q)))^2}{\sum_{k\in K}(U^{km})^2}) \tag{37}$$

Rearranging the terms lead to the above formula. □

This form of constraint, however, might be too conservative (i.e., the true probability is much lower than the desired value κ^m). de Nijs et al. (2017) propose a dynamic update approach by replacing the value C^m in constraints (34) with a more relaxed limit $\overline{C}^m \geq C^m$. First, the problem is solved to optimality with C^m and a simulation is run to evaluate the true probability of constraint violation and to determine the value, denoted by $\hat{C}^m$, where the probability of constraint violation equals the desired value κ^m based on the empirical estimate of the distribution. Then a value of $\overline{C}^m$ within the range $C^m \leq \overline{C}^m < \hat{C}^m$ is chosen for constraints (34) used in the next iteration. The process continues until the probability of constraint violation based on the empirical estimate is sufficiently close to the desired value κ^m. Since constraints (34) are always relaxed in the subsequent iterations, this approach guarantees that all solutions generated during the process are feasible and the algorithm can terminate early if necessary.

5 Numerical experiments

We perform extensive numerical experiments to demonstrate the benefits of the proposed approach which can be used with generic discrete lead-time demand functions to determine inventory policies for multiple items under multiple inventory targets. Section 5.1 presents the cross-validation process that evaluates the quality of different inventory policies, and Section 5.2 details the generation of the datasets based on the literature. Section 5.3 evaluates the performance and scalability of the proposed optimization approaches. Section 5.4 investigates the performance of the approach to estimating the probability distributions and their impact on the resulting policy. Finally, Section 5.5 compares our approach with the method proposed in Downs et al. (2001), where the authors define base stock policies for a multi-item inventory optimization problem with a global constraint. This last section also investigates the impact of the forms of the inventory policy (i.e, stochastic lookup table, deterministic lookup table, rule-based) on the inventory management performance.

The models are solved with CPLEX 20.2, and the algorithms are implemented in Julia. The code is available at `https://github.com/Simon-Thevenin/Multi-Item_Inventory_Planning_A_Mathematical_Programming_Approach`. Each test is run on one core of a processor Intel Xeon Brodwell EP E5-2630v4 with 10 cores, a frequency of 2.20GHz, and 128GB of RAM memory. We limit the RAM memory usage per test to 10 GB. In the experiments, we set the maximum inventory level equal to two times the maximum demand, and we compute $Q^k(i)$ in such a way that it respects the inventory level.

5.1 Cross-validation

To assess the quality of the inventory policies obtained by the optimization approaches, we simulate their performances on test data that haven't been used to generate the inputs for the models. In our experiments, we split the data into two sets based on the temporal order of observations. The set with the older observations is used to determine (optimize) the policies, and the set with more recent observations is used for cross-validation. We explain how the data is generated and split temporally into a training set and a testing (holdout) set in the subsequent sections which present the details of different datasets used in the experiments.

The execution of the policy depends on its type. The execution of the policy is straightforward for approaches that result in an (s, S) policy based on its inventory position (Axsäter 2015). When the solution of the model comprises the probability $x^k_{i,q}$ (which corresponds to an order quantity q to be made when the inventory position equals i for item k), we create a policy lookup table to store all the values of $\hat{x}^k_{i,q}$ which represent the probability to order q when the inventory position is i. In the case of deterministic policy, there is only a single quantity decision q associated with a given state i. Note that this is a pessimistic simulation: no policy change is allowed during the simulation.

5.2 Datasets

This paper provides methods applicable to a wide range of inventory management situations. In the experiments, we focus on two types of situations. Instances of type A correspond to the case where a company seeks a policy that minimizes production and inventory costs to meet a global service level target. The costs include fixed ordering costs O^k, unit costs o^k, and holding costs h^k. The minimum stockout probability for each item is set to 90%, and an average stockout probability over all the items is set to 95%. The instance set B is similar to the ones used in Downs et al. (2001). In these instances, the company aims to balance inventory and lost sale costs while respecting a maximum inventory constraint. We consider a capacity on the inventory level set to 75% of the full capacity, where the full capacity corresponds to the news vendor level. Table 4 details the distribution used to generate the cost parameters for both instance types.

In both types of instances, we consider two demand profiles, namely, slow mover (SM) and fast mover (FM). We use the demand generator presented in Petropoulos et al. (2014). To simulate intermittent data, three parameters are used: the average inter-demand interval (IDI); the squared coefficient of variation of positive demands; and the number of observations. The inter-demand interval is based on a Bernoulli distribution with $p = \frac{1}{IDI}$, while the positive demands follow a negative binomial distribution. This dataset was generated using the R code provided in Petropoulos et al. (2014). The SM (resp. FM) demands are generated using a squared co-variance of 2 (resp. 1.4) and an inter-demand interval average of 4 resp (1). The average demand is randomly selected in the interval $[10, 50]$ for FM items, and it is set to 5 for SM items. The number of periods in the historical data set varies from 36 to 208, and this corresponds to 6 months up to 4 years of historical data. The simulation data set contains a large number of periods (1000) to reduce the variance in the simulation. Finally, we assume the lead times follow a uniform discrete distribution with support $tildeL^k = \left[L^k_{min}, L^k_{max}\right]$. We select the values of L^k_{min} and L^k_{max} randomly in the interval $[5, 20]$ or $[1, 3]$.

Table 4: Data generation

Parameter	Instance A	Instance B
Lead times min L^k_{min}	uniform(5,20)	uniform(5,20)
Lead times max L^k_{max}	uniform(L_{min},20)	uniform(5,20)
Holding cost h_k	1	uniform(5,20)
Lost sales e_k	0	uniform(10,20)
Unit cost v_k	uniform(1,10)	0
Fixed cost s_k	Computed based on the time between order (TBO) from EOQ formula $s = \frac{\bar{D}hTBO^2}{2}$	0
Time Between Order (TBO)	uniform(1,20)	-

5.3 Performance of the CMDP-based method

This section evaluates the numerical performance of the proposed methods. We first investigate the performance of exact methods that find a deterministic policy. The paper provides three such approaches: (1) the MILP (7)–(13) (denoted by MILP), (2) the Branch and Price approach (denoted by BP-LP) presented in Section 3.4 where the sub-problem is an LP, (3) the Branch and Price approach (denoted by BP-sS) presented in section 4.1 where the solution of the sub-problem takes the form of an (s, S) policy. We evaluate these methods on instances of set A. As we focus on the computational efficiency of the method in this section, we only consider the empirical distribution built from 52 periods of historical data. Table 5 reports the computation time (CPU) and the gap with the best solution (GAP) for the solutions obtained with MILP, BP-LS, and BP-sS. The GAP is computed as $GAP = \frac{f_M - \text{Best}}{\text{Best}}$, where Best is the cost of the best solution, and f_M is the cost of the solution obtained with method M. In addition, Table 5 gives the number of action-states of the instance. The number of action-states is directly related to the size of the model, and it is computed by:

$$\sum_{k \in K} \sum_{i \in I^k} |Q_k(i)|.$$

Finally, Table 5 reports the relative integrality gap (Rel. GAP) reported by CPLEX for MILP, and the number of columns (#Columns) generated by the branch and price approaches.

Table 5: Performance of the solution approach to the deterministic policy model

						MILP			BP-LP			BP-sS		
aggreg	# item	Item type	Lead time range	# action-state	Best	GAP (%)	CPU (s)	Rel Gap(%)	GAP (%)	# Columns	CPU (s)	GAP (%)	# Columns	CPU (s)
1	10	SM	(1, 3)	24,979.7	231.8	0.0	65.4	0.0	0.0	795.0	3606.1	0.1	389.0	1389.6
		SM	(5, 20)	82,825.8	1806.8	0.0	142.5	0.0	0.0	257.0	3610.1	0.0	19.0	276.1
		FM	(1, 3)	1,154,718.7	-	-	-	-	-	-	-	-	-	-
	100	SM	(1, 3)	272,103.9	3069.7	-	-	-	0.0	51.0	3618.9	0.2	29.0	3630.1
		SM	(5, 20)	852,517.6	-	-	-	-	-	-	-	-	-	-
		FM	(1, 3)	25,901,228.2	-	-	-	-	-	-	-	-	-	-
	1000	SM	(1, 3)	2,914,510.5	-	-	-	-	-	-	-	-	-	-
		SM	(5, 20)	7,460,006.4	-	-	-	-	-	-	-	-	-	-
		FM	(1, 3)	114,683,150.0	-	-	-	-	-	-	-	-	-	-
10	10	SM	(1, 3)	13,034.5	480.8	0.0	29.2	0.0	0.0	10000.0	3600.0	0.0	1603.0	85.3
		SM	(5, 20)	5,094.5	2086.6	0.0	19.4	0.0	0.0	142.0	34.7	0.0	15.0	23.4
		FM	(1, 3)	12,768.2	5466.5	0.0	23.4	0.0	0.0	7.0	28.2	0.0	7.0	28.8
	100	SM	(1, 3)	280,281.6	5387.2	0.0	3620.5	0.4	1.9	3139.0	3600.3	1.4	7071.0	3600.3
		SM	(5, 20)	55,283.0	20623.4	0.0	3623.3	0.3	1.2	1492.0	3728.0	0.8	2593.0	3756.8
		FM	(1, 3)	115,259.7	71439.1	0.1	3702.4	0.3	0.0	106.0	3619.6	0.0	81.0	3601.7
	1000	SM	(1, 3)	1,289,365.0	59180.3	0.3	3669.6	1.4	2.4	13.0	3741.2	2.2	100.0	3600.3
		SM	(5, 20)	203.0	184,771.4	42.1	6308.1	42.8	0.5	12.0	3840.8	0.3	72.0	3644.8
		FM	(1, 3)	455.8	480,235.7	-	-	-	0.0	3.0	3637.7	0.6	3.0	4028.3
20	10	SM	(1, 3)	3,746.6	760.1	0.0	21.4	0.0	0.0	468.0	33.2	3.2	754.0	35.9
		SM	(5, 20)	2,333.1	2442.6	0.0	20.9	0.0	0.3	5469.0	3601.7	0.1	308.0	34.4
		FM	(1, 3)	4667.2	5816.4	0.0	28.4	0.0	0.0	1260.0	1145.8	0.0	404.0	178.1
	100	SM	(1, 3)	76,068.0	8215.4	0.0	3620.9	0.4	3.0	4986.0	3600.6	4.7	10000.0	3600.0
		SM	(5, 20)	24079.1	24275.7	0.0	3621.1	0.2	1.2	2756.0	3600.6	0.9	6966.0	3600.3
		FM	(1, 3)	43010.5	74570.5	0.0	3640.0	0.1	0.0	357.0	3606.6	0.0	354.0	3602.5
	1000	SM	(1, 3)	365,883.0	88,817.8	0.0	3638.9	1.4	5.9	15.5	3650.5	5.9	100.0	3600.1
		SM	(5, 20)	43,010.5	220,492.3	0.0	3672.1	0.6	2.3	14.5	3780.7	2.2	100.0	3600.1
		FM	(1, 3)	365,883.0	499,542.8	-	-	-	0.2	6.5	3842.4	0.0	18.0	3652.2
Average						2.4	2192.6	2.7	0.9	1492.8	3025.1	1.1	1475.5	2360.4

Table 5 shows that the MILP can solve all the instances with up to 100 items and the instances with 1000 items for the demand type SM when an aggregation factor of at least 10 is applied to reduce the size of the model. Nevertheless, the MILP model could not find a solution for some instances with 1000 because the model is too large, and the branch-and-bound tree could not even fit in the memory. Branch-and-price approaches perform better than MILP since they can solve all instances with an aggregation factor of more than 10. However, exact methods (MILP and branch-and-price) could solve only a few instances without state aggregation (i.e., aggregation factor equals one). For these instances, the model is so large that even the decomposition methods cannot solve it. Note that the branch-and-price approach could yield high-quality solutions where the average optimality gap equals 0.9% and 1.1% for BP-LS and BP-sS, respectively. On the contrary, when MILP does not solve an instance to optimality, the GAP can still be large. Note also that BP-sS requires fewer iterations than BP-LS when both approaches converge to an optimal solution. For instance, for the instances with 10 slow mover items, an aggregation factor of 1, and a lead time range $[5, 20]$, BP-sS converges in 19 iterations on average versus 257 iterations for BP-LS.

As exact approaches cannot solve instances with a very large action-state space, we investigate the performance of heuristics methods. These approaches include (1) the column generation (denoted by CG) approach presented in section 3.4 where the sub-problem is an LP, and the integer master problem is solved after generating the columns on the version with relaxed integrality, (2) the column generation approach (denoted by CG-sS) where the sub-problem takes the form of an sS policy as presented in Section 4.1, and (3) the improvement of CG-sS (denoted by CG-LS) where the sub-problem is solved with local search and sampling as presented in Section 4.2.

Table 6 reports the GAP and CPU for these methods, as well as the results of MILP for comparison. The results show that the column generation approaches could find solutions for more instances than MILP. In addition, the average GAP for CG-LS and CG-sS is similar to the one of MILP, but the computation time is smaller. Note that CG-LS is able to solve all instances since it relies on sampling to evaluate a solution, and it does not require generating the entire state space. The GAP of CG-LS is computed based on the sampling approximation, and it can slightly deviate from the true cost of the resulting policy. Nevertheless, CG-LS obtains high-quality policies with an average GAP of only 1.1%.

We now investigate the performances of approaches when stochastic (non-deterministic) policies are employed. These methods include the linear program (LP) (7)–(11), and the column generation (CG) approaches. Table 7 reports the results in the same format as Table 5. The results show that for non-deterministic policy, the LP approach outperforms the column generation method since it obtained the same solution faster. Some instances have such a large number of action-states per item that the model remains too large even when it is decomposed per item. For other instances, building and solving the model takes a few seconds (1-10), and CG could not generate the column required to find a feasible solution within the time limit of one hour.

From these analyses, we derive the following insights into the performance of the methods. Decision makers seeking a deterministic policy for less than 100 items and with an aggregation factor of 10 or more should use MILP. Branch-and-price approaches are slower than MILP for these instances where the state-action space is not large. However, the BP-LP approach is much more efficient when the inventory management involves more than 1000 items whereas the BP-sS is recommended when the decision maker requires the policies to take the form of an (s, S) policy. CG-LS is the most appropriate method for situations when the state-action space is very large (e.g., more than 10^6 in size). Even though the CG method does not guarantee to converge to the optimal solution, its optimality gap can be derived and our numerical experiments demonstrate that the final optimality gap is approximately only 1%.

5.4 Performance comparisons based on different demand distributions

This section investigates the impact of the lead-time demand distributions on the performance of the policy. We use the instance of set A where the distribution is learned from historical data of different sizes (32, 52, 104, 208 periods), and the resulting policy is simulated over a large number $(10,000)$ of

Table 6: Performance of heuristic approach to the deterministic policy model

aggreg	# item	Item Type	Lead time range	# action-state	Best	MILP GAP (%)	MILP CPU (s)	CG-LP GAP (%)	CG-LP CPU (s)	CG-sS Exact GAP (%)	CG-sS Exact CPU (s)	CG-LS GAP (%)	CG-LS CPU (s)
1	10	SM	(1, 3)	24979.7	231.8	0.0	65.4	0.1	37.7	0.1	41.8	-2.3	3606.3
		SM	(5, 20)	82825.8	1806.8	0.0	142.5	0.0	119.0	0.1	146.6	-1.0	3611.2
		FM	(1, 3)	1154718.7	5130.7	-	-	-	-	-	-	0.0	3668.5
	100	SM	(1, 3)	272103.9	3069.7	-	-	0.0	1218.5	0.3	2051.9	2.1	3843.8
		SM	(5, 20)	852517.6	18081.1	-	-	-	-	-	-	0.0	3818.0
		FM	(1, 3)	25901228.2	76820.2	-	-	-	-	-	-	0.0	5601.9
	1000	SM	(1, 3)	2914510.5	36246.0	-	-	-	-	-	-	0.0	3780.6
		SM	(5, 20)	7460006.4	158646.4	-	-	-	-	-	-	0.0	3787.8
		FM	(1, 3)	114683150.0	469215.4	-	-	-	-	-	-	0.0	3678.7
10	10	SM	(1, 3)	475.2	480.8	0.0	29.2	4.5	24.0	4.5	22.0	6.7	3600.2
		SM	(5, 20)	1253.6	2086.6	0.0	19.4	0.9	22.5	0.7	22.4	-1.2	3601.8
		FM	(1, 3)	13034.5	5466.5	0.0	23.4	0.0	18.0	0.0	28.3	-1.4	3608.1
	100	SM	(1, 3)	5094.5	5387.2	0.0	3620.5	3.9	32.9	3.8	32.0	6.2	3634.4
		SM	(5, 20)	12768.2	20623.4	0.0	3623.3	1.5	41.8	1.1	34.9	-0.1	3665.1
		FM	(1, 3)	280281.6	71439.1	0.1	3702.4	0.0	405.7	0.0	427.3	0.0	3687.7
	1000	SM	(1, 3)	55283.0	59180.3	0.3	3669.6	2.4	1968.3	2.4	335.2	3.3	3801.4
		SM	(5, 20)	115259.7	184771.4	42.1	6308.1	0.5	2802.4	0.3	2197.4	0.4	3814.8
		FM	(1, 3)	1289365.0	480235.7	-	-	0.0	4469.6	0.7	4033.2	-1.3	3704.9
20	10	SM	(1, 3)	203.0	760.1	0.0	21.4	10.0	23.7	10.0	23.2	10.5	3601.8
		SM	(5, 20)	455.8	2442.6	0.0	20.9	2.0	23.9	2.0	8.3	-1.3	3600.3
		FM	(1, 3)	3746.6	5816.4	0.0	28.4	0.0	25.4	0.0	9.6	-1.4	3600.3
	100	SM	(1, 3)	2333.1	8215.4	0.0	3620.9	7.3	31.5	7.2	28.7	8.8	3606.9
		SM	(5, 20)	4667.2	24275.7	0.0	3621.1	2.2	34.5	1.8	30.8	-0.3	3641.7
		FM	(1, 3)	76068.0	74570.5	0.0	3640.0	0.1	149.5	0.1	105.4	-0.3	3788.8
	1000	SM	(1, 3)	24079.1	88817.8	0.0	3638.9	6.3	1460.6	6.3	181.6	1.5	3649.5
		SM	(5, 20)	43010.5	220492.3	0.0	3672.1	2.3	1688.7	2.3	276.7	0.4	3647.1
		FM	(1, 3)	365883.0	499542.8	-	-	0.1	3840.7	0.0	2240.9	1.0	3754.0
			Average			2.4	2192.6	2.1	878.0	2.1	584.7	1.1	3755.8

demand and lead time samples. We also analyze the impact of the aggregation factor on the cost of the resulting policy in the simulation.

Figures 2a (resp. 2b) shows the percentage difference between the simulation costs of the policies associated with the empirical (resp. zero-inflated negative binomial) distribution inferred from different aggregation factors and history size for fast moving items. For instance, “Emp-26” provide the GAP when an empirical distribution is built with 26 periods of historical demand. Figures 2c and 2d show the same results for slow-moving items. The results show that the aggregation factor is a sensitive parameter when the demand is modeled with an empirical distribution. A small aggregation factor improves the performance of the solution process, but a too-large aggregation factor could lead to poor results. However, the aggregation factor has a negative impact when the SM demand is modeled with a parametric distribution. When demand is modeled with an empirical distribution, the cost of the policy decreases when the aggregation factor increases until a certain level, and then the cost increases when the aggregation factor passes this optimal level. The optimal value of the aggregation factor depends on the type of demand and on the size of the historical data. For instance, for fast movers, the optimal value of the aggregation factor is 10 with 6 months of history and 2 with 4 years of history. For slow movers, larger values of the aggregation factor may be used, and it seems that an aggregation factor of 10 leads to good results for all historic sizes.

When demand is represented using an empirical distribution, the positive impact of the aggregation factor is due to the reduction of the number of parameters, and the resulting model does not over-fit the data with a larger bin size. The aggregation factor is less important for slow movers because there are fewer demand values to infer the distribution (as most demands are 0).

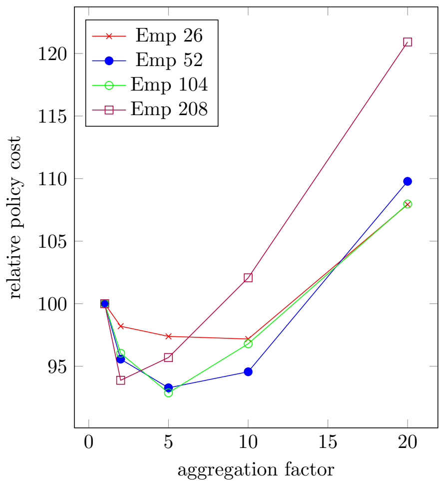
Table 7: Comparison of the solution approach to the non-deterministic policy model

aggreg[redacted]	# item	Item Type[redacted]	Lead time range	# action-state	Best	LP GAP (%)	LP CPU (s)	CGLP GAP (%)	CGLP CPU (s)
1	10	SM	(1, 3)	24979.7	231.8	0.0	21.9	0.0	37.7
		SM	(5, 20)	82825.8	1806.7	0.0	58.6	0.0	119.0
		FM	(1, 3)	1154718.7	-	-	-	-	-
	100	SM	(1, 3)	272103.9	3042.1	0.0	130.2	0.0	1218.5
		SM	(5, 20)	852517.6	-	-	-	-	-
		FM	(1, 3)	25901228.2	-	-	-	-	-
	1000	SM	(1, 3)	2914510.5	-	-	-	-	-
		SM	(5, 20)	7460006.4	-	-	-	-	-
		FM	(1, 3)	114683150.0	-	-	-	-	-
10	10	SM	(1, 3)	475.2	466.2	0.0	18.5	0.0	24.0
		SM	(5, 20)	1253.6	2086.4	0.0	18.7	0.0	22.5
		FM	(1, 3)	13034.5	5466.5	0.0	21.1	0.0	18.0
	100	SM	(1, 3)	5094.5	5324.1	0.0	19.9	0.0	32.9
		SM	(5, 20)	12768.2	20559.4	0.0	22.1	0.0	41.8
		FM	(1, 3)	280281.6	71350.3	0.0	143.4	0.0	405.7
	1000	SM	(1, 3)	55283.0	58430.2	0.0	64.4	0.0	1968.3
		SM	(5, 20)	115259.7	182597.0	0.0	127.4	0.0	2802.4
		FM	(1, 3)	1289365.0	467644.9	0.0	2246.3	0.4	4469.6
20	10	SM	(1, 3)	203.0	739.8	0.0	19.7	0.0	23.7
		SM	(5, 20)	455.8	2435.4	0.0	20.7	0.0	23.9
		FM	(1, 3)	3746.6	5814.9	0.0	19.9	0.0	25.4
	100	SM	(1, 3)	2333.1	8044.6	0.0	19.9	0.0	31.5
		SM	(5, 20)	4667.2	24143.5	0.0	20.7	0.0	34.5
		FM	(1, 3)	76068.0	74480.9	0.0	39.1	0.0	149.5
	1000	SM	(1, 3)	24079.1	87135.6	0.0	37.0	0.0	1460.6
		SM	(5, 20)	43010.5	219141.4	0.0	55.2	0.0	1688.7
		FM	(1, 3)	365883.0	498574.3	0.0	386.0	0.0	3840.7
Average						0.0	167.2	0.0	878.0

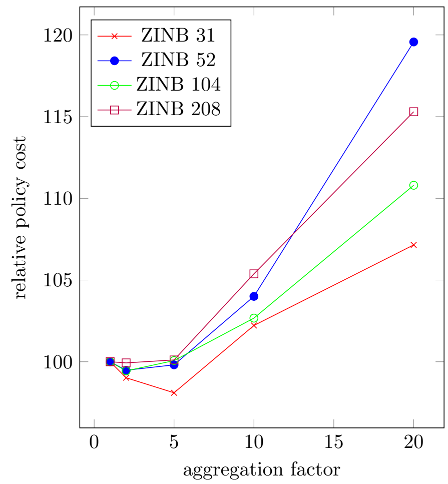
5.5 Computational comparisons with discretization approach for base-stock policy adapted from Downs et al. (2001)

This section evaluates the performance of various inventory policies. We compare the stochastic policy obtained by solving model (7)–(11), the deterministic policy obtained from model (7)–(13), and the reorder point policy obtained from CG with an (s, S) policy. For each of these policies, we consider the case where the demand distribution function is an empirical distribution (EMP) or a zero-inflated negative binomial (ZINB), and the cases with no demand aggregation (aggregation factor of 1) or an aggregation factor of 5. We also compare the approaches proposed in this paper with the approach proposed in Downs et al. (2001).

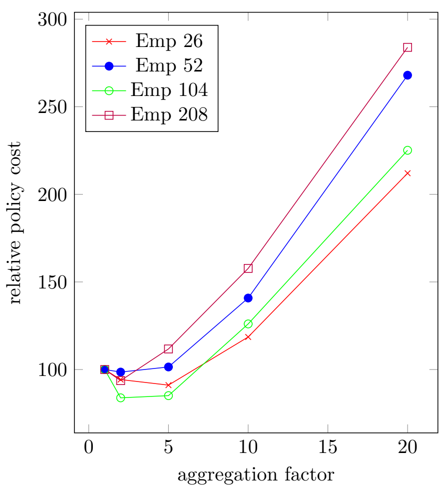
We first compare our approach to the model adapted from the framework presented in Downs et al. (2001) which can be used to optimize the base stock inventory policies (i.e., a single base stock level S) based on discrete demand functions. The main purpose of this section is to provide computational insights on the solutions (policies) produced by our approach and the approach adapted Downs et al. (2001) rather than demonstrating the superiority of one solution approach over the other. Even though the two approaches were developed to deal with generic discrete demand functions, they differ in several ways, i.e., (i) the inventory model of Downs et al. (2001) is developed for a base stock policy where no fixed charge is considered and the resource constraint is imposed only on the base stock level i, (ii) the cost function used in the original framework of Downs et al. (2001) is calculated based on a single *replay* scenario over a long term horizon which is based on the demand in the training set, whereas our CMDP models rely on a transition function generated from a discrete distribution, and (iii) the model in Downs et al. (2001) implicitly considers lags in delivery whereas our CMDP model allows a single



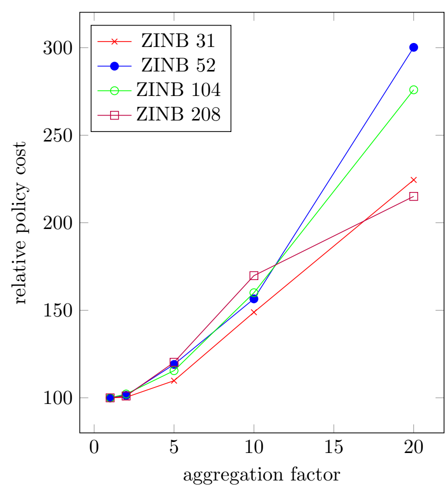
(a) Fast moving items with EMP distribution.



(b) Fast moving items with ZINB distribution.



(c) Slow moving items with EMP distribution.



(d) Slow moving items with ZINB distribution.

Figure 2: Relative cost in the simulation of the policy computed for different demand types and different distributions derived from a history of 32, 52, 104, or 208 weeks and different aggregation factors

replenishment to be made for each item during the supply lead time. As presented in Downs et al. (2001), we use the scenario that corresponds to the historical demand to calculate the cost parameters of the model. In addition, since the model of Downs et al. (2001) can only be used to determine base stock policies, we consider only the inventory holding and stockout costs in this numerical experiment and the policy determined by the CMDP model takes the form of a base stock policy in this case. The details of the model adapted from Downs et al. (2001) are described in Appendix 3.

In this section, we consider instances of type B, and we rely on equation (34) to constrain the maximum inventory level, where we set the upper limit κ on the probability of violating the maximum inventory constraint to 0.01. The maximum resource consumption per item at an instant is the largest value of the inventory level i. We present the performance of the approaches, using the cross-validation technique shown in Section 5.1.

Table 8 reports the cross-validation results in the format (Avg cost, const, 95% const, 90% const), where Avg is the average cost in the simulation, const is the proportion of the instances where the maximum inventory is below the limit, 95% const is the proportion of instances where the 95^{th} percentile inventory level is below the inventory limit, and 90% const is the proportion of instances where the 90^{th} percentile of the inventory level is below the limit. No results are reported for lookup policies with an aggregation factor of 1 for fast mover because this leads to a too large model to fit in memory.

Table 8: Costs and the percentage of instances where the maximum inventory constraint is satisfied based on different approaches

Policy	Aggregation Factor	Distribution	**Slow Movers**	**Fast Movers**
MIP-BS adapted from Downs et al. (2001)	n/a	n/a	(138.52, 1, 1, 1)	(3222.43, 1, 1, 1)
Stochastic policy lookup table	1	EMP	(148.57, 0.2, 0.4, 1)	-
		ZINB	(134.77, 1, 1, 1)	-
	5	EMP	(144.49, 1, 1, 1)	(3842.66, 1, 1, 1)
		ZINB	(144.21, 1, 1, 1)	(2187.67, 1, 1, 1)
Deterministic policy lookup table	1	EMP	(149.05, 0.2, 0.6, 0.8)	-
		ZINB	(134.87, 1, 1, 1)	-
	5	EMP	(144.48, 1, 1, 1)	(3975.78, 1, 1, 1)
		ZINB	(144.84, 1, 1, 1)	(2197.43, 1, 1, 1)
(s, S)	1	EMP	(136.63, 0.4, 0.8, 1)	(2335.49, 1, 1, 1)
		ZINB	(134.77, 0.8, 1, 1)	(2220.73, 0.6, 1, 1)
	5	EMP	(142.91, 1, 1, 1)	(2213.82, 1, 1, 1)
		ZINB	(144, 1, 1, 1)	(2208.28, 1, 1, 1)

Table 8 shows that the zero-inflated negative binomial distribution yields better policy than the empirical distribution. This can be explained by the ability of the ZINB to generalize the data while Emp overfits by definition. Thus the use of ZINB reduces over-fitting. An issue with the empirical distribution is missing values. Some value of the distributions are not part of the samples, this lead to some state with zero probability in the policy, and thus no action is associated with these states. In the simulation, when a state with no action is encountered, we perform a random action. This leads to poor results in the simulation.

However, the (s^k, S^k) policy is less sensitive to the bad approximation provided by the empirical. For instance, for the slow mover with an aggregation factor of 1, the cost of the (s^k, S^k) policy decrease from 136.63 to 134.77 when the distribution changes from EMP to ZINB, whereas these cost increase from 134.77 to 149.05 with a deterministic lookup policy. A possible reason is that the (s^k, S^k) policy is simpler since there are fewer parameters to optimize. As a result, the values of s^k and S^k can be well selected despite poor distributional information, whereas the lookup policy optimizes a complex policy over the wrong distribution and this could lead to poor results.

The results show that the approach proposed in this paper yields lower average costs than the approach of Downs et al. (2001). The first reason is that the approach of Downs et al. (2001) imposes a strict limit on the constraint where the approaches proposed in this paper ensure meeting the constraint with probability 99%. In addition, the approach of Downs et al. (2001) determines the inventory policy decisions based on previous historical demand (replay method) and thus the solutions can potentially overfit the input data and the results based on cross-validation are not very robust as opposed to the CMDP approaches. Finally, we note that the gap between different methods can be large for fast-moving items. This is related to the issues of the unseen demand values, which is exacerbated because there are only 52 observed demand values but there can be a large number of possible demand values for fast-moving items. We want to further note that the standard MILP or LP approach is much more computationally demanding in terms of the memory required as opposed to the local search approach used to solve the subproblem when it is restricted to an (s, S) policy. As a consequence, the standard approach cannot provide results for fast-moving items without demand aggregation.

5.6 Computational results on real-world industrial dataset

We also test our CG heuristic using a real-world dataset obtained from an industrial partner. The details and descriptive statistics on the industrial dataset are presented in Appendix. The experiments were performed using the parameters and service targets provided by the industrial partner. In a standard setting of their inventory optimization tool, it generates (s, S) policies using either Poisson or Normal distribution which is pre-assigned to each group of items where the batch quantity B to set the level $S = s + B$ is calculated based on a standard EOQ calculation (Axsäter 2015). This benchmark approach (based on either Gaussian or Poisson distribution) is indicated by the label P_{ith}. Based on their current approach, to select the policies that satisfy the predefined target levels, the tool generates 12 inventory policy choices based on different service levels (percentiles of lead-time demand distribution) for each item. These policies are then used to create the decision variables for policy selection. To choose the policies that collectively satisfy the target levels, they solve a set-partitioning model using a set of policies generated a priori for each item. Based on the service requirements specified by the partner, only fill rate (FR(%)) is considered as the global target level. In terms of the CMDP approach, we employed the CG-LS approach using the empirical distribution (Emp) and the zero-inflated negative binomial distribution (ZINB).

In Table 9, the term *CSL(%)* indicates the cycle service level or no-stockout probability, where *FR(%)* indicates the fulfillment rate. We also report other statistics from the cross-validation including relative total cost (*T.Cost*), and relative inventory value (*Inv. Val.*). Under the column(s) associated with target(s) used in each dataset, we report the expected values obtained by solving the optimization models in column *Exp*, the values obtained by the cross-validation process in column *CV* and the absolute difference between *Exp* and *CV* in column $|Diff|$.

Table 9: Cross-validation results on industrial dataset.

			FR(%)			Cross-Validation Results		
Group	Approach	Policy generation	Exp	CV	|Diff|	T.Cost	Inv. Val.	CSL (%)
B1	CMDP	LS(Emp)	95.0	94.4	**0.6**	1.032	1.077	90.8
	CMDP	LS(ZINB)	95.0	90.2	4.8	1.000	1.246	94.3
	MIP	P_{ith}(Normal)	95.0	87.2	7.8	1.137	1.019	91.5
	MIP	P_{ith}(Poisson)	95.0	83.3	11.7	1.164	1.000	91.1
B2	CMDP	LS(Emp)	95.0	94.2	**0.8**	1.032	1.094	91.5
	CMDP	LS(ZINB)	95.0	91.4	3.6	1.000	1.314	95.0
	MIP	P_{ith}(Normal)	95.0	88.0	7.0	1.233	1.017	92.0
	MIP	P_{ith}(Poisson)	95.0	84.2	10.8	1.265	1.000	91.6
B3	CMDP	LS(Emp)	95.0	94.1	**0.9**	1.026	1.085	92.0
	CMDP	LS(ZINB)	95.0	90.7	4.3	1.000	1.281	95.2
	MIP	P_{ith}(Normal)	95.0	87.4	7.6	1.185	1.015	92.3
	MIP	P_{ith}(Poisson)	95.0	83.9	11.1	1.213	1.000	91.8
B4	CMDP	LS(Emp)	95.0	93.7	**1.3**	1.049	1.093	92.4
	CMDP	LS(ZINB)	95.0	89.8	5.2	1.000	1.239	95.0
	MIP	P_{ith}(Normal)	95.0	87.3	7.7	1.188	1.018	92.4
	MIP	P_{ith}(Poisson)	95.0	84.2	10.8	1.216	1.000	91.9
B5	CMDP	LS(Emp)	95.0	93.9	**1.1**	1.015	1.169	92.5
	CMDP	LS(ZINB)	95.0	89.8	5.2	1.000	1.245	95.1
	MIP	P_{ith}(Normal)	95.0	86.6	8.4	1.147	1.031	92.2
	MIP	P_{ith}(Poisson)	95.0	82.7	12.3	1.185	1.000	91.6
Average	CMDP	LS(Emp)	95.0	94.0	**1.0**	1.031	1.104	91.9
	CMDP	LS(ZINB)	95.0	90.4	4.6	1.000	1.265	94.9
	MIP	P_{ith}(Normal)	95.0	87.3	7.7	1.178	1.020	92.1
	MIP	P_{ith}(Poisson)	95.0	83.7	11.3	1.209	1.000	91.6

Table 9 shows the results based on the cross-validation on this dataset. We can see, for the benchmark approaches P_{ith}(Normal) and P_{ith}(Poisson), that such distributions cannot properly represent the demand distributions of irregular items used in the experiments since the fill rates, based on the cross-validation results, are significantly different from the expected value, which then satisfies the target constraint. As we have seen in the first experiment, both CMDP methods perform relatively well. However, the method CMDP-LS(Emp) is generally better than that of CMDP-LS(ZINB) in terms of the difference between the expected value (in-sample) and target level based on cross-validation as shown in column $|Diff|$ whereas the CMDP-LS(ZINB) approach produces solutions with lowest expected total costs among these methods.

6 Conclusion

This paper presents a multi-item inventory optimization approach that makes use of the coupled, constrained Markov decision process (CMDP) to determine inventory policy decisions under multiple target levels. We presented the models, decomposition framework, as well as computational enhancements. To demonstrate the performance of the approach, the computational comparisons with other approaches in the literature, including the discretization approach for inventory planning with base stock policy and generic discrete demand distributions on the synthetic and industrial datasets of slow-moving products, are provided. The results show that the proposed approaches, in conjunction with the uses of empirical or zero-inflated distribution, could generally capture the demand profiles of irregular items such as slow-moving products and determine the inventory policies. These could collectively satisfy multiple target levels and result in smaller gaps between the expected values (in-sample) and cross-validation results compared to other approaches.

References

Akçay, A., Biller, B., and Tayur, S. R. (2016). Beta-guaranteed joint service levels. Working Paper, SSRN: https://ssrn.com/abstract=2709937.

Altman, E. (1999). Constrained Markov decision processes, volume 7. CRC Press.

Axsäter, S. (2015). Inventory control, volume 225. Springer.

Bello, D. and Riano, G. (2006). Linear programming solvers for Markov decision processes. In IEEE Systems and Information Engineering Design Symposium, pages 90–95.

Beyer, D., Sethi, S. P., and Sridhar, R. (2001). Stochastic multiproduct inventory models with limited storage. Journal of Optimization Theory and Applications, 111(3):553–588.

Boutilier, C. and Lu, T. (2016). Budget allocation using weakly coupled, constrained Markov decision processes. In Proceedings of the Thirty-Second Conference on Uncertainty in Artificial Intelligence (UAI-2016), pages 52–61. AUAI Press.

Chang, H. S., Hu, J., Fu, M. C., and Marcus, S. I. (2013). Simulation-based algorithms for Markov decision processes. Springer Science & Business Media.

Chapados, N. (2014). Effective bayesian modeling of groups of related count time series. In Proceedings of The 31st International Conference on Machine Learning, pages 1395–1403.

Cheng, F. and Sethi, S. P. (1999). A periodic review inventory model with demand influenced by promotion decisions. Management Science, 45(11):1510–1523.

Choi, J., Cao, J. J., Romeijn, H. E., Geunes, J., and Bai, S. X. (2005). A stochastic multi-item inventory model with unequal replenishment intervals and limited warehouse capacity. IIE Transactions, 37(12):1129–1141.

Chu, P., Yang, K.-L., and Chen, P. S. (2005). Improved inventory models with service level and lead time. Computers & Operations Research, 32(2):285–296.

Dantzig, G. B. and Wolfe, P. (1960). Decomposition principle for linear programs. Operations research, 8(1):101–111.

de Nijs, F., Walraven, E., de Weerdt, M. M., and Spaan, M. T. (2017). Bounding the probability of resource constraint violations in multi-agent mdps. In AAAI, pages 3562–3568.

DeCroix, G. A. and Arreola-Risa, A. (1998). Optimal production and inventory policy for multiple products under resource constraints. Management Science, 44(7):950–961.

Derman, C. and Veinott Jr, A. F. (1972). Constrained Markov decision chains. Management Science, 19(4-part-1):389–390.

Desaulniers, G., Desrosiers, J., and Solomon, M. M. (2006). Column generation, volume 5. Springer Science & Business Media.

Dolgui, A. and Pashkevich, M. (2008). Demand forecasting for multiple slow-moving items with short requests history and unequal demand variance. International Journal of Production Economics, 112(2):885–894.

Downs, B., Metters, R., and Semple, J. (2001). Managing inventory with multiple products, lags in delivery, resource constraints, and lost sales: A mathematical programming approach. Management Science, 47(3):464–479.

Ehrhardt, R. (1984). (s, S) policies for a dynamic inventory model with stochastic lead times. Operations Research, 32(1):121–132.

Federgruen, A. and Zipkin, P. (1984). An efficient algorithm for computing optimal (s, S) policies. Operations research, 32(6):1268–1285.

Federgruen, A. and Zipkin, P. (1986). An inventory model with limited production capacity and uncertain demands I. the average-cost criterion. Mathematics of Operations Research, 11(2):193–207.

Graves, S. C. and Willems, S. P. (2000). Optimizing strategic safety stock placement in supply chains. Manufacturing & Service Operations Management, 2(1):68–83.

Hoeffding, W. (1963). Probability inequalities for sums of bounded random variables. Journal of the American statistical association, 58(301):13–30.

Ignall, E. and Veinott Jr, A. F. (1969). Optimality of myopic inventory policies for several substitute products. Management Science, 15(5):284–304.

Johnson, E. L. (1967). Optimality and computation of (σ, s) policies in the multi-item infinite horizon inventory problem. Management Science, 13(7):475–491.

Kelle, P. (1989). Optimal service levels in multi-item inventory systems. Engineering Costs and Production Economics, 15:375–379.

Ketzenberg, M., Metters, R., and Semple, J. (2006). A heuristic for multi-item production with seasonal demand. IIE Transactions, 38(3):201–211.

Long, D. Z., Sim, M., and Zhou, M. (2023). Robust satisficing. Operations Research, 71(1):61–82.

McKinsey & Company (2019). Turn slow-moving inventory into fast profits. https://www.mckinsey.com/business-functions/operations/our-insights/turn-slow-moving-inventory-into-fast-profits. (Accessed on 06/07/2020).

Minka, T. P. (2001). A family of algorithms for approximate Bayesian inference. PhD thesis, Massachusetts Institute of Technology.

Petropoulos, F., Makridakis, S., Assimakopoulos, V., and Nikolopoulos, K. (2014). 'Horses for Courses' in demand forecasting. European Journal of Operational Research, 237(1):152–163.

Puterman, M. L. (2014). Markov decision processes: discrete stochastic dynamic programming. John Wiley & Sons.

Scarf, H. (1960). The optimality of (S, s) policies in the dynamic inventory problem. Mathematical Methods in the Social Sciences, pages 196–202.

Thonemann, U. W., Brown, A. O., and Hausman, W. H. (2002). Easy quantification of improved spare parts inventory policies. Management Science, 48(9):1213–1225.

Tiwari, V. and Gavirneni, S. (2007). ASP, the art and science of practice: Recoupling inventory control research and practice: guidelines for achieving synergy. Interfaces, 37(2):176–186.

Veinott, Jr, A. F. (1966). On the opimality of (s,S) inventory policies: New conditions and a new proof. SIAM Journal on Applied Mathematics, 14(5):1067–1083.

Veinott Jr, A. F. (1965). Optimal policy for a multi-product, dynamic, nonstationary inventory problem. Management Science, 12(3):206–222.

Wang, X. (2011). Continuous review inventory model with variable lead time in a fuzzy random environment. Expert systems with applications, 38(9):11715–11721.

Yin, K., Liu, H., and Johnson, N. E. (2002). Markovian inventory policy with application to the paper industry. Computers & Chemical Engineering, 26(10):1399 – 1413.

Zhang, X., Meiser, D., Liu, Y., Bonner, B., and Lin, L. (2014). Kroger uses simulation-optimization to improve pharmacy inventory management. Interfaces, 44(1):70–84.

Zipkin, P. (1986). Stochastic leadtimes in continuous-time inventory models. Naval Research Logistics Quarterly, 33(4):763–774.

Zipkin, P. (2008). On the structure of lost-sales inventory models. Operations research, 56(4):937–944.

Zuur, A. F., Ieno, E. N., Walker, N. J., Saveliev, A. A., and Smith, G. M. (2009). Zero-truncated and zero-inflated models for count data. In Mixed effects models and extensions in ecology with R, pages 261–293. Springer.