Response to Review Report of Paper Titled "Single-site Perishable Inventory Management under Uncertainties: A Deep Reinforcement Learning Approach" (Paper ID = TKDE-2022-05-0714)

Dear TKDE Associate Editor and Reviewers,

First, we would like to thank you for your constructive and inspiring comments. We have gone through these comments carefully and revised our draft accordingly. Below we summarize the major changes.

1) We have moved related work to Section 2, added Table 1 to demonstrate the differences between our work and the related studies and included more detailed elaborations on these studies.
2) We have listed our assumptions of the problem setting after the problem definition in Section 3.
3) We have included more discussions and explanations on the rationale of capturing open orders, lead time prediction and demand prediction for defining states in Section 4.1.
4) We have moved the details of the hyperparameter selection and model training, the main results on real dataset and synthetic dataset to Section 5.2.
5) We have added an experiment on varying the maximum time period T, and included the results and discussions in a technical report [1].
6) We have put the proof of the competitive analysis, the offline algorithm and some additional experimental results in a technical report [1].

Appended please find our detailed responses to all reviewers' comments. We really appreciate your feedback and time.

Yours sincerely,
Kaixin, Cheng, Darrell, Jie, and Xue-Ming

Comments and Suggestions from Review #1

1. *The abstract should be improved. I would suggest adding specific information as a conclusion statement.*

Response: We followed the suggestion and included some statistics of the efficiency and effectiveness improvement in the abstract.

2. *The Related Work section of the manuscript is poor.*

   a. *Related to the subjects and methods discussed in the study, additional references should be provided.*
   b. *Furthermore, the superiority and distinctive aspects of this study should be emphasized compared to the existing studies in the literature.*
   c. *This section can be rearranged as Section 2 after the introduction section.*

Response: We followed the suggestion and made the following changes in the revised manuscript. First, we added a table, i.e., Table 1, to demonstrate the differences between our work and related studies. Second, we included more detailed elaborations on these related studies. Third, we re-arranged the related work section as Section 2 after the introduction section.

3. *In Section 2 and 3, the problem description and methodology are very well expressed.*

Response: Thanks for your valuable comment. In the revised manuscript, we included a summarization of our problem settings in Section 3 (Section 2 of the original version) and more discussions on the rationale behind the designs in Section 4 (Section 3 of the original version). We believe these updates would help to improve the presentation of the problem description and methodology.

4. *What will it provide to add lead time and demand predictions to the state information of the problem? Please discuss this in detail.*

Response: We included more discussions regarding the definitions of the lead time s_i^l and demand predictions s_i^d in Section 4.1. We note that these two variables both capture important information that is useful for planning the orders, and the effectiveness of introducing them is verified to some extent by the results of the ablation study on the state and action definitions, provided in Section 5.2.

5. *The authors should further explain the open orders in the state information.*

Response: We included more discussions on the definition of the open orders. Specifically, it captures the quantity of materials that would arrive at the inventory in the following time periods, which is important for planning the future orders.

6. *Information about the deep learning networks in the RL4LS algorithm should be presented in Section 3.3.*

Response: We included the detailed settings of the networks and the details of how they are trained in Section 5.1, together with the specifications of the hyperparameter selection.

7. *Assumptions about the problem should be listed.*

Response: We followed the suggestions and summarized the settings of the problem just after the definition. Specifically, the settings include (1) we target an *online* planning setting; (2) we target the *lost-sale* scenario, where the customers would not wait for the stock to be replenished and drop the unfilled demands; (3) the planning is in a *periodic-review* manner.

8. *It should be explicitly indicated that the requirements and limitations of the presented method.*

Response: We followed the suggestion and indicated the limitations of the proposed method in the conclusion section. We copy the discussions here for easier reference. It is worthy of noting that our proposed solution is not suitable for the materials whose associated costs vary significantly in different time periods since those costs are considered consistent in our problem. Also, since we do not consider a fill rate (fr), i.e., the ratio of the satisfied demands to the total demands, in our model. If the inventory has such a goal, e.g., $fr \geq 95\%$, our solution would not guarantee to achieve it.

9. *The strengths of the algorithm presented in the manuscript can be analyzed from different perspectives.*

Response: The strengths of our algorithm are in threefold. First, consider the effectiveness. Compared with several heuristic methods [2], [3], our proposed method could

capture the essential information from the environment, including inventory level, open orders, lead time and demands, adaptively due to its data-driven nature. Compared with other RL-based algorithm, we proposed a more powerful state representation, which could extract richer information from the environment. Second, consider the efficiency. The time complexity of our method is linear wrt the maximum time period T if we have a well-trained model. This would be efficient for real inventory planning. Third, the training process of our model is cheap. As indicated in Section 5.2, we can train a satisfactory model within 2 hours. We have covered the discussions on these strengths in Section 1 and Section 5.

10. *The additional experimental results section should be moved to section 4.2 and section 4.2 should be enriched.*

Response: **We followed the suggestion and moved most of the main results to Section 5.2.**

11. *In the conclusion section, it should be stated the limitations of the proposed model.*

Response: **We followed the suggestion and indicated the limitations of the proposed method in the conclusion section.**

12. *English language, although suitable, need to be correct for some typos and errors in the manuscript.*

Response: **Thank you for your careful checking. We have fixed these typos in the revised version and further proofread the revised paper carefully.**

Comments and Suggestions from Review #2

1. *The appendix contains much important content and should be put in the main body of the paper. Especially, for the ablation study of the experiments, it can show much details of the performance of the proposed algorithm.*

Response: **We followed the suggestion and moved most of the main results to Section 5.2.**

2. *Reinforcement learning is suitable to solve this kind of online optimizing problems. I agree the choice to use the RL framework. However, my suggestion is to discuss more on the advantage of the proposed algorithm in this paper. I notice in the existing studies, other algorithms also use RL. Thus, why the proposed algorithm in this paper can outperform the existing RL algorithms? Maybe more intuitive and theoretical analyses can make the paper more solid.*

Response: **Existing studies [4], [5], [6], [7], which use RL methods for inventory planning, differ from our study in various aspects. We have pointed out the differences in the related work section explicitly. For some related RL-based methods (which target different settings), namely QL4LS and PG4LS, we adapted them to our problem. However, they are outperformed by the methods proposed in this paper. One of the reasons is that our state definition captures some essential information that is not captured by QL4LS and PG4LS. Furthermore, we conducted extensive ablation studies on combining different state and action definitions (including those of QL4LS and PG4LS) to show the superiority of our solution (of state and action definitions), which can be found in Section 5.2.**

3. *The experiments are in kind of small scales. In both real and synthetic datasets, there are only 12 periods, which is too small in my opinion. Try to enlarge the number of periods. Also add more experiments to study the effects of other important parameters, such as the number of materials, suppliers and so on.*

Response: **We followed the suggestion by conducting the experiments on some large $T \in \{50, 100, 150, 200, 250, 300\}$ and including the experimental results on synthetic dataset in the technical report [1]. We also varied the number of the materials and suppliers following the existing study [8] and reported the results in Figure 5.**

Comments and Suggestions from Review #3

1. *In the MDP, each state is defined as a $(2L + |S|)$-dimension vector, where L is the maximum number of time periods and $|S|$ is the number of suppliers. In cases with many time periods and suppliers, would the state space be too large, and the model be difficult to train? The authors are suggested to provide discussions on this potential issue.*

Response: **We note that L is the maximum life time of the material, which is different from the maximum number of time periods T. For perishable materials, such as chemicals, food and medicine, they usually do not have a very long life time L. Also, in real scenarios, a company often has tens of the suppliers for a certain materials, which makes it feasible to train a model.**

2. *In experiments, for both the real dataset and synthetic dataset, $T = 12$ time periods are involved. This looks to be quite limited. For example, with this small number of time periods, would the data be sufficient to train reliable models? The authors should provide some discussions and justify the reliability of the results for the real datasets.*

Response: **We followed the suggestion by conducting experiments on some large $T \in \{50, 100, 150, 200, 250, 300\}$ and including the experimental results on synthetic dataset in the technical report [1].**

3. *Would the real datasets be made publicly available? The authors are encouraged to publish the datasets used in this paper (especially the real one from the industry) so as to benefit the community working on similar problems. The authors are also encouraged to publish their codes if possible.*

Response: **Thank you for your valuable comment. We provided a link of our code in the experiment section. We are sorry that we cannot make the real dataset publicly available as it contains commercial information of a company.**

Single-site Perishable Inventory Management under Uncertainties: A Deep Reinforcement Learning Approach

Kaixin Wang, Cheng Long, Darrell Joshua Ong, Jie Zhang, and Xue-Ming Yuan

Abstract—Online lot sizing for perishable materials in an uncertain environment is a fundamental problem for inventory planning and has been studied in the past several decades. In this paper, we study a novel setting of the lot sizing problem, considering perishable materials, multiple suppliers, uncertain demands and lead time (LS-PMU), which captures the inventory planning task in real life better than existing lot sizing problems. We present theoretical results of the best possible competitive ratio an online algorithm can achieve for LS-PMU problem. We then develop a reinforcement learning based algorithm called RL4LS to intelligently choose the supplier and decide the order quantity in each time period. We conduct extensive experiments on both real and synthetic datasets to verify that RL4LS outperforms existing algorithms in terms of effectiveness and efficiency, e.g., RL4LS improves the effectiveness by 44% and runs two orders of magnitude faster than the state-of-the-art algorithm IBFA.

Index Terms—supply chain optimization, inventory management, lot sizing, deep reinforcement learning

1 Introduction

Supply chain management is to manage the flow of raw materials, semi-finished and finished products. During the process of supply chain management, one essential task is *inventory planning*, which involves a critical problem called *lot sizing* [9]. Specifically, in a lot sizing problem, a decision-maker needs to determine in each period the orders to be placed and their quantities, considering the lead time (which means the time needed for producing and transporting the material from the supplier to the inventory), associated costs, products' quality, to minimize the total costs. Depending on the application scenarios, different assumptions, objectives, and constraints are adopted for the lot sizing problem [10]. Most of existing works [6], [11], [12] for the lot sizing problem assume a single-supplier scenario, where there is only one supplier available across different periods. More recent works [2], [3] target a multiple-supplier scenario. Specifically, [2], [3] assume that the decision-maker can order materials from *multiple* suppliers in each *single* period. We observe that this multiple-supplier scenario does not capture the practice that well. For example, for many enterprises, such as ST Logistics, Alcon, LSH Electrical Engineering, they always view the order in a period as an integrated one and place it to the most suitable supplier, but not multiple ones for the following reasons.

First of all, it would save the ordering costs. Most suppliers charge a fixed cost, i.e., a setup cost, in addition to the costs of purchasing the materials, which usually cover the expenses incurred in production and shipment. Therefore, when we distribute the orders to multiple suppliers in a period, we may not only pay more money for buying these materials (since different suppliers charge different unit costs), but also have a higher expenditure for the production and transportation from different suppliers. Second, it would introduce less work of coordination when receiving orders. Imagine that we order from multiple suppliers in every period. Then for some periods, we may need to handle the situation that a number of receiving orders from different suppliers arrive at the inventory simultaneously, which makes the coordination a tough job. Third, it would make it convenient for after-sale services. When the customers want to return or exchange their products due to some reasons, it would be easier to contact the supplier.

In this paper, we propose a new problem, called the lot sizing with perishable materials, multiple suppliers and uncertain demands and lead time (LS-PMU). We adopt the setting that one only places the order of the material to at most one supplier in each planning period, which is different from that adopted in existing studies [2], [3].

Consider a planning scenario of T periods with a set of suppliers, denoted by S. Existing solutions usually adopt a vector-based solution, which represents the ordering policy as a $|S|$-dimensional vector in each planning period [2], [3], but they apply different strategies to find the ordering policy with the minimum total costs. One state-of-the-art method is [2], called GA. This method applies the genetic algorithm to find the optimal solution, where a possible solution is encoded as a chromosome-like data structure, represented by a $(|S| \times T)$-dimensional vector. Another method is proposed in [3], called IBFA. This method applies an improved bacteria foraging algorithm to find the best solution, where the raw solutions, intermediate results and the global optimum are all demonstrated by $(|S| \times T)$-dimensional vectors. These existing solutions, however, cannot handle our task well since we do not allow to place orders to multiple suppliers

- K. Wang, C. Long and J. Zhang are with the School of Computer Science and Engineering, Nanyang Technological University, Singapore Email: {kaixin.wang, c.long, zhangj}@ntu.edu.sg.
- D. J. Ong and X. Yuan are with Singapore Institute of Manufacturing Technology, Singapore. Email: {darrell_joshua_ong, xmyuan}@simtech.a-star.edu.sg.

TABLE 1
Related literature of lot-sizing problems.

Reference	Material	Setting	Deterioration	Supplier	Method[1]
[4], [5], [13], [14], [15], [16], [17], [18], [19]	Non-perishable	*[2]	N.A.	*	*
[12], [20], [21], [22], [23], [24], [25], [26], [27]	Perishable	Offline	*	*	DP / Heuristics
[28], [29], [30], [31]	Perishable	Online	Continuous	*	Simulation / Heuristics
[6], [32], [33], [34]	Perishable	Online	Fixed Lifetime	Single	SP / Heuristics / RL
[2], [3]	Perishable	Online	Fixed Lifetime	Multiple	Nature-inspired Heuristics
Ours	Perishable	Online	Fixed Lifetime	Multiple[3]	RL

[1] DP represents dynamic programming; SP represents stochastic programming; RL represents reinforcement learning.
[2] We use * to denote that all types under the criteria (column) are covered by some of the references of the row.
[3] Our problem targets a different multiple-supplier setting compared with [2], [3].

in a single period. In addition, [2], [3] are both based on the nature-inspired heuristics, and it would take a long time for them to converge.

To address the LS-PMU problem effectively and efficiently at the same time, we consider the planning process of LS-PMU problem as a sequential decision making process. Then, we model the sequential decision making process as a Markov Decision Process (MDP) [35]. We carefully design the MDP including states, actions and rewards such that (1) the states capture the critical information of the inventory status and cheap to compute; (2) the actions are *hybrid* ones and capture the decision process of LS-PMU (i.e., first choosing a supplier, which is captured by a discrete value, and then deciding an order quantity, which is a continous value); and (3) the rewards are well aligned with the goal of the problem. Finally, we adopt an existing reinforcement learning algorithm, namely parameterized Deep-Q Network (P-DQN) [36], to learn the policy for the MDP. In summary, the main contributions of this paper are as follows.

- We propose the LS-PMU problem, in which we place orders to at most one supplier in each period, which is more aligned with real logistics applications.
- We present theoretical results on the lower bound of the competitive ratio of any given algorithm for LS-PMU problem, which could reflects the best possible quality guarantee for any online algorithm to the LS-PMU problem.
- We propose a reinforcement learning based algorithm RL4LS for the LS-PMU problem. Compared with existing methods to the lot sizing problem, our algorithm can capture more information of the inventory status and as a result, can generate the policies with smaller costs.
- We conduct extensive experiments on both real and synthetic datasets to verify that RL4LS outperforms all (resp. most) existing algorithms in terms of effectiveness (resp. efficiency). For example, RL4LS improves the effectiveness by 44% and simultaneously runs 226 times faster on the real dataset, compared with the state-of-the-art IBFA.

The rest of this paper is organized as follows. We provide the problem definition and theoretical results in Section 3. We present our RL4LS algorithm in Section 4 and show the experimental results in Section 5. We summarize the related work in Section 2. Finally, we conclude our paper in Section 6.

2 Related Work

Lot-sizing Problems. Lot sizing problems can be classified based on several criteria as presented in Table 1. Among them, [4], [5], [13], [14], [15], [16], [17], [18], [19] target non-perishable materials. For perishable materials, earlier studies on the lot-sizing problem assume an *offline* setting, where the demands in each period are known in advance [12], [20], [21], [22], [23], [24], [25], [26], [27]. Besides, most of them assume that the lead time is always zero. Recently, there exist some studies [6], [28], [29], [30], [31], [32], [33], [34] which consider the online setting. However, [28], [29], [30], [31] target the materials whose quality deteriorates continuously over time, i.e., $N(t) = N(0) \cdot e^{-\lambda t}$, where $N(0)$ and $N(t)$ are the initial quality and the quality at time t, respectively. [6], [32], [33], [34] target the materials that have a fixed expiry date, but they assume a *single* supplier. The most related studies are [2], [3] since they also target perishable materials with multiple suppliers. Nevertheless, as explained in Section 1, these studies allow to place orders to more than one suppliers in a single period, which do not capture some real scenarios well. Interested readers are referred to a survey paper [10] for more details of other types of lot-sizing problems.

Reinforcement Learning. Given a specific environment, which is generally formulated as a Markov Decision Process (MDP) [35], reinforcement learning is to help the agents in the environment learn how to map the states to actions so as to maximise the accumulative rewards [37]. Some existing studies [4], [5], [6], [7] have tried to solve the inventory planning problem using reinforcement learning, they differ from our RL4LS algorithm in various aspects: (1) [4], [5] allow to place orders to more than one suppliers in a single period, and they focus on non-perishable materials; (2) [6] assumes zero or constant lead time; (3) [7] targets a multiple-echelon scenario that multiple inventories cooperate to minimize the total costs or maximize the revenue.

3 Problem Statement

3.1 Variables

We consider a horizon of time periods $1, 2, ..., T$, where each corresponds to a unit of time such as a day, a week, or a month. Let S be a set of suppliers, from which we can order materials. During each time period, the material of some quantity may be involved in different events, e.g., it is ordered from some supplier, delivered to the inventory, sold to customers, and disposed. We define the following variables to represent the quantities involved in different events.

We denote by $o(i, j)$ and $l(i, j)$ the quantity and lead time of the material ordered from supplier $j \in S$ at the beginning of time period $i \in [1, T]$, respectively. Here, the lead time $l(i, j)$ means the amount of time it takes for the

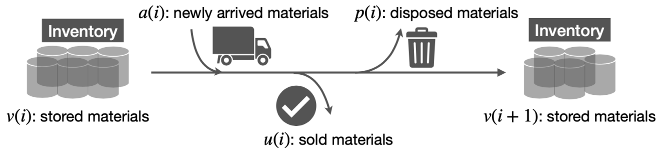


Fig. 1. An illustration of the variables and events between two adjacent time periods.

ordered material to be transported from the supplier j to the inventory starting at the beginning of time period i. $l(i,j)$ usually depends on the various transportation factors such as weather and is unknown for future time periods. Note that the order $o(i,j)$ will be received at the beginning of the time period $i' = i + l(i,j)$.

We denote by $v(i)$ the quantity of the material stored in the inventory at the beginning of time period i. We further define $v(i,r)$ to be the quantity of material which (1) is stored in the inventory at the beginning of time period i and (2) has the remaining life time $r \in [1, L]$, where L is the maximum life time of the material. Note that we have $v(i) = \sum_{r=1}^{L} v(i,r)$. We assume that the initial inventory level is zero, i.e., $v(1) = 0$.

We denote by $a(i)$ the quantity of the material that arrives at the inventory at the beginning of time period i. We assume that no material expires before it arrives at the inventory, which is well aligned with the practice. Thus, this quantity depends on how the material is ordered and transported before time period i, i.e., $o(i',j)$ and $l(i',j)$ for $i' \le i$ and $j \in S$. We also define $a(i,r)$ to be the quantity of material which (1) arrives the inventory at period i and (2) has the remaining life time $r \in [1, L]$. Similarly we have $a(i) = \sum_{r=1}^{L} a(i,r)$.

We denote by $d(i)$ the quantity of the material that is demanded by customers during time period i. This quantity for the future time periods is unknown. We further denote by $s(i)$ the quantity of the material that is in shortage during time period i. This quantity is equal to $\{d(i) - v(i) - a(i)\}^+$, where $\{x\}^+ = \max\{0, x\}$. Here, we target the lost-sales scenario, where the customers would drop the unfilled demands incurring some shortage cost, which will be introduced in the next subsection.

Besides, we denote by $u(i)$ the quantity of the material that is used to serve customers' demands during the time period i. Note that we have $u(i) = \min\{d(i), v(i) + a(i)\}$.

Finally, we denote by $p(i)$ the quantity of the material that is disposed at the end of time period i. In this case, this quantity depends on the quantity of material that will expire at the end of this time period, i.e., $v(i,1) + a(i,1)$, and the customer demand during this time period, i.e., $d(i)$. Specifically, we have $p(i) = \{v(i,1) + a(i,1) - d(i)\}^+$. Here, we adopt the principle that material with shorter remaining life time is used to serve customers' demands first (i.e., First-Expired-First-Out (FEFO)).

Note that the quantity of the material at the beginning of the next time period $i+1$, i.e., $v(i+1)$, depends on $v(i)$ (existing), $a(i)$ (newly arrived), $u(i)$ (sold), and $p(i)$ (disposed). Specifically, we have the following equation.

$$v(i+1) = v(i) + a(i) - u(i) - p(i). \quad (1)$$

An illustration of the above events is shown in Figure 1.

3.2 Costs

Due the life time of the material, various costs are incurred, including ordering cost, holding cost, disposal cost and shortage cost for each time period i. We denote these costs by $C_o(i)$, $C_h(i)$, $C_d(i)$, and $C_s(i)$, respectively. Let $c_o(j)$, c_h, c_d, and c_s be the unit cost of ordering from supplier j, holding, disposing, and having a shortage of the material, respectively. Let $c_b(j)$ be the cost that would be incurred when a certain quantity of material is ordered from the supplier j, i.e., the fixed setup costs. Then, the costs can be computed as follows.

$$\begin{aligned} C_o(i) &= \sum\nolimits_{j\in S}(c_o(j)\cdot o(i,j) + c_b(j)\cdot I(o(i,j))) \\ C_h(i) &= c_h \cdot \{v(i) + a(i) - d(i)\}^+ \\ C_d(i) &= c_d \cdot \{v(i,1) + a(i,1) - d(i)\}^+ \\ C_s(i) &= c_s \cdot \{d(i) - v(i) - a(i)\}^+ \end{aligned} \quad (2)$$

where $I(x)$ is an indicator function, which is equal to 1 if $x > 0$ and 0 otherwise.

Then, the cost for time period i, which we denote by $C(i)$, can be computed as follows.

$$C(i) = C_o(i) + C_h(i) + C_d(i) + C_s(i) \quad (3)$$

3.3 Problem Definition

In this paper, we study the problem of lot sizing with perishable materials, multiple suppliers, and uncertain demands and lead time (LS-PMU). Specifically, the problem is to decide for each time period a supplier and a quantity for ordering a material in an online fashion so as to minimize the total cost over all time periods subject to the inventory capacity constraint and the principle that during each time period, we order the material from at most one supplier only. Mathematically, the problem could be formalized as follows.

$$\min_{o(i,j), i\in[1,T], j\in S} \quad \sum\nolimits_{i=1}^{T} C(i) \quad \text{s.t.} \quad (4)$$

$$v(i) \le V \text{ for each } i \in [1,T] \quad (5)$$

$$\sum\nolimits_{j\in S} I(o(i,j)) \le 1 \text{ for each } i \in [1,T] \quad (6)$$

where V is the capacity for the material and $I(x)$ is an indicator function, which is equal to 1 if $x > 0$ and 0 otherwise. In addition, we summarize settings of our problem. (1) We target an *online* planning setting; (2) we target the *lost-sale* scenario, where the customers would not wait for the stock to be replenished and drop the unfilled demands; (3) the planning is in a *periodic-review* manner.

3.4 Competitive Ratio Analysis

In the following, we present a competitive ratio boundary that an algorithm is able to achieve for the LS-PMU problem.

Theorem 1. *Suppose that $\min_{j\in S} c_o(j) \le c_s$ (since otherwise the competitive ratio would be exactly 1). Given an algorithm $A \in \mathcal{A}$ for the LS-PMU problem with decision variables $o(i,j)$ for all i and j. Define*

$$o(i) = \sum_{j\in S} o(i,j) \quad (7)$$

and

$$cr(i) = \max\left(\frac{(c_h + c_d)\cdot o(i) + f(o(i))}{f(0)}, \frac{c_s}{\min\limits_{j\in S}\{c_o(j)\}}\right), \tag{8}$$

where

$$f(x) = \min_{j\in S}\{c_o(j)\cdot x + c_b(j)\}, \quad x \geq 0 \tag{9}$$

then the competitive ratio of algorithm A is at least $\min_i cr(i)$.

We provide the proof of Theorem 1 and also an offline optimal algorithm A^* in the technical report [1]. With A^*, we can compute the empirical competitive ratio of an algorithm. Theorem 1 provides the bounds under the worst-case, but in practice, we observe that the ratio of the total costs returned by our algorithm to the total costs returned by A^* is only around 2.1 on our real datasets.

4 METHODOLOGY

We observe that the inventory planning task in the LS-PMU problem corresponds to a sequential decision process, i.e., it makes decisions on (1) choosing a supplier and (2) determining the order quantity for placing an order on the chosen supplier in each period sequentially. Therefore, we propose to use reinforcement learning to help with the decision making process. Specifically, we model the LS-PMU problem as a Markov Decision Process (MDP), adopt an existing deep reinforcement learning method, namely P-DQN [36], for learning an optimal policy on the MDP, and then develop an algorithm called RL4LS, which uses the learned policy for solving the LS-PMU problem.

4.1 The LS-PMU Problem Modeled as an MDP

We model the LS-PMU problem as an MDP, which mainly consists of three components, namely states, actions, and rewards as defined as follows.

States. We denote the state in time period i by s_i. Intuitively, the state s_i should capture essential information of the inventory status for making decisions of choosing a supplier and setting the order quantity in a period. We identify the following three types of information at the beginning of the time period i, which are critical for making the decisions: (1) the inventory level, (2) open orders, which have been ordered but not received by the inventory and (3) the prediction of the lead time and demands.

To capture the first type of information of the inventory status at the beginning of time period i, we take the values $v(i, r)$ for $r \in [1, L]$ and concatenate them together, denoted by s_i^v, that is,

$$s_i^v = [v(i,1), \cdots, v(i,r), \cdots v(i,L)] \tag{10}$$

The rationale is that s_i^v provides a clear overview of the materials that can be directly consumed in the current time period.

To capture the second type of information of the inventory status, we maintain a list of open orders $\mathcal{O}$. Based on the assumptions that (1) no material expires before they arrive at the inventory and (2) we order the material from at most one supplier in each period, there would be at most $L-1$ open orders at the beginning of the time period i. Let $b(k)$ be a binary value, where $b(k) = 1$ indicates the order placed at time period $k < i$ is an open order and $b(k) = 0$ otherwise. Then, we use s_i^o, as defined below, to capture the open order information,

$$s_i^o = [b(i-L+1)\cdot o(i-L+1), \cdots, b(i-1)\cdot o(i-1)] \tag{11}$$

where $o(\cdot) = \sum_{j\in S} o(\cdot, j)$. The rationale is that s_i^o clearly captures the quantity of materials that would arrive at the inventory in the following time periods, which is important for planning the future orders. Note that for those $r \in [1, L-1]$ with $i - r \leq 0$, we assume the open orders are equal to 0.

Finally, to capture the third type of information of the inventory status, we make some predictions on the lead time and the demands. Firstly, the supply lead time is stochastic and usually varies among different suppliers. Moreover, we may choose different suppliers in different periods. Therefore, lead time prediction of a supplier j provides the information of how long it will take to deliver the materials if we place the order to j, which would affect the decision afterwards. Formally, based on the ordering history of different suppliers, we predict the lead times for different suppliers, which are denoted by $\hat{l}(i,j)$ for $j \in S$. Then, we concatenate the predicted lead times for different suppliers and denote it by s_i^l,

$$s_i^l = [\hat{l}(i,1), \cdots, \hat{l}(i,j), \cdots \hat{l}(i,|S|)], \quad j \in S \tag{12}$$

Similarly, demand prediction provides the information of how many materials we need to meet in the following time period, which would further affect the choice of the supplier since if there is a surge demand, we need to choose the supplier with a short lead time. Formally, the predicted demand, denoted by $\hat{d}(i)$, and the corresponding state representation s_i^d are defined as follows.

$$s_i^d = [\hat{d}(i)] \tag{13}$$

We try different methods for lead time and demand prediction, e.g., ARIMA [38], LSTM [39] and sampling from known distribution. We compare the results and report them in the experiments. In summary, we define state s_i to be a $(2L + |S|)$-dimension vector, which captures the inventory level information, open order information and the future predictions, i.e.,

$$s_i = [s_i^v, s_i^o, s_i^l, s_i^d] \in \mathbb{R}^{2L+|S|} \tag{14}$$

Actions. We denote the action at time period i by a_i. Recall that at the beginning of the time period i, the state is s_i and we need to decide (1) how much quantity we order and (2) which supplier we choose from. Formally, we define a_i as follows.

$$a_i = [j, q] \quad (j \in S) \tag{15}$$

which means we place an order to the supplier j in which the quantity of the material is q. We note that the actions defined above are *hybrid* ones, which involve both a discrete value j (which indicates a supplier) and a continuous value q (which means the order quantity).

Rewards. Consider that we take action a_i at a state s_i and then we arrive at a new state s_{i+1}. We define the reward associated with this transition from s_i to s_{i+1}, denoted by R_i, as follows.

$$R_i = -C(i) \tag{16}$$

The intuition is that if a smaller cost is incurred in period i, the action a_i would be associated with a larger reward, and vice versa. It is worthy of mentioning that with the reward defined as above, the objective of the MDP, which is to maximize the accumulative rewards, would be equivalent to that of the LS-PMU problem, which is to minimize the costs over all periods. To see this, suppose that we go through a sequence of states $s_1, s_2, \cdots, s_T$ and correspondingly we receive a sequence of rewards R_1, R_2, R_{T-1}. Then, the accumulative rewards without being discounted can be computed by

$$\sum_{i=1}^{T} R_i = -\sum_{i=1}^{T} C(i). \tag{17}$$

4.2 Policy Learning on the MDP

Considering that our MDP has a high dimensional continuous state space and a hybrid action space, we adopt P-DQN [36], which is capable of dealing with continuous states and hybrid actions, to learn a policy from our MDP. P-DQN involves a Q network and a policy network μ. The Q network, parameterized by ω, is to estimate the action-value function, and the policy network μ, parameterized by θ, is to map the state to an order quantity given a supplier in a deterministic way. The μ network given a supplier j is denoted by μ_j. Thus, given a state s_i in period i, we can select a proper action by first choosing a supplier j^* based on the Q network, and then calculating the order quantity q^* based on the μ networks, i.e.,

$$\begin{aligned} j^* &= \arg\max_{j \in S} Q(s_i, j, \mu_j(s_i;\theta);\omega) \\ q^* &= \mu_{j^*}(s_i;\theta), \end{aligned} \tag{18}$$

We then take the action $a_i = [j^*, q^*]$ in period i.

For the training process, we initialize two main networks $Q(\cdot;\omega)$ and $\mu(\cdot;\theta)$, which are used for selecting actions, and two target networks $Q'(\cdot;\omega')$ and $\mu'(\cdot;\theta')$, which are used for calculating the losses for training the main networks. During the training, we adopt the ϵ-greedy method, which takes the action $a = [j^*, q^*]$ with a probability of $(1-\epsilon)$ and a random action $a = [j, q]$ other than $[j^*, q^*]$ with a probability of ϵ, to balance exploration and exploitation. We also maintain a replay memory, which contains the latest several transitions that are used for training the networks. Then, the training process is as follows. Consider N experiences sampled uniformly from the replay memory, i.e., (s_i, a_i, R_i, s_{i+1}) for $i = 1, 2, \cdots N$. For the Q network, we compute the loss by

$$L(\omega) = \frac{1}{N}\sum_{i=1}^{N}\Big(y_i - Q(s_i, a_i;\omega)\Big)^2 \tag{19}$$

where

$$y_i = R_i + \gamma \cdot \max_{j' \in S} Q'(s_{i+1}, j', \mu'_{j'}(s_{i+1};\theta');\omega'). \tag{20}$$

For the policy network μ, we compute the loss by

$$L(\theta) = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{|S|} Q(s_i, j, \mu_j(s_i;\theta);\omega). \tag{21}$$

Algorithm 1 The RL4LS algorithm

Require: The demands $d(i)$ and the lead time $l(i,j)$ for all i and j, where they arrive in an online fashion;
Ensure: The ordering plan $o(i,j)$ for all i, j, total costs C;

1: Initialize the open order list $O \leftarrow \phi$;
2: $C \leftarrow 0$;
3: **for** each time period i where $i \in [1, T]$ **do**
4: $s_i \leftarrow$ observe the inventory status as Eq. (14);
5: Take an action $a_i = [j^*, q^*]$ which satisfies Eq. (18);
6: Output $o(i, j^*) = q^*$ and $o(i,j) = 0$ for $j \neq j^*$;
7: $C(i) \leftarrow$ total costs in the time period i;
8: $C \leftarrow C + C(i)$;
9: **end for**
10: **return** C

Finally, we update the parameters θ and ω by gradient descent.

4.3 The RL4LS Algorithm

The RL4LS algorithm, presented in Algorithm 1, is based on the learned policy for the MDP that models the LS-PMU problem. Specifically, it proceeds periodically (line 3). At each time period i, it observes the inventory status based on the three types of information, and constructs a state s_i (line 4). Then, it takes an action $a_i = [j^*, o^*]$ based on the learned networks $Q(s, a;\omega)$ and $\mu(s;\theta)$ (line 5-6). Then, the ordering plan for the time period i will be (1) $o(i, j^*) = q^*$ and (2) $o(i,j) = 0$ for other supplier j. After that, we observe the costs and move to the next period (lines 7-8).

Time complexity. The time complexity of the RL4LS algorithm is $O(T(L + |S|))$, where T denotes the maximum planning horizon and L is the maximum life time. The cost mainly comes from the transition of the states. Given a state $s = [s^v, s^o, s^l, s^d]$, we analyze the costs as follows. For s^v and s^o, they both take $O(L)$ time to construct. For s^l and s^d, they can be calculated in $O(|S|)$ and $O(1)$ time, respectively. Therefore, the complexity of the RL4LS is $O(T(L + |S|))$.

5 Experiment

5.1 Experiment Setup

Datasets. The real dataset is collected by Alcon Singapore Manufacturing Pte. Ltd [40]. There are 76 different materials, and for each material, there are 76 different suppliers. Different materials have different demand distributions, maximum life time and associated costs. And different suppliers also have different lead time distributions. The planning horizon is one year. Since orders are placed on a monthly basis, we set $T = 12$. The dataset contains some private commercial information, thus, we omit the detailed descriptions of this dataset in this paper. We also generate some synthetic datasets by following the existing study [8], which considers a sequence T of 12 periods and use different settings of $|S|$ for generating suppliers. For each setting, we further generate the same number of materials. Details of generating the suppliers and materials are referred to [8].

Baselines and Metrics. We compare our proposed model with four existing algorithms, namely IBFA [3], GA [2], PG4LS [4] and QL4LS [6], in terms of the total cost and

TABLE 2
Running time (seconds) on real dataset.

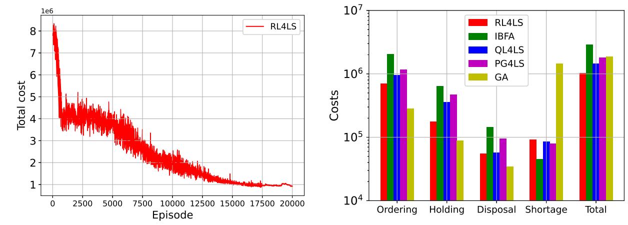
Alg.	RL4LS	IBFA	QL4LS	PG4LS	GA
Time	2.39	542.27	1.78	2.77	843.74

the running time. IBFA and GA are two state-of-the-art algorithms for the lot-sizing problem with perishable materials under the multiple-supplier setting, which use two different nature-inspired heuristics. PG4LS is the most recent reinforcement learning based algorithm for non-perishable inventory planning under the multiple-supplier setting. These three algorithms are all vector-based solutions, i.e., they allow to place orders to more than one supplier in a single period. We adapt them to our problem by the following steps. Suppose the output for a period i can be represented by a vector $[p_1, p_2, \cdots, p_{|S|}]$. First, we set the order quantity to $q_i = \sum_{j=1}^{|S|} p_j$. Second, we choose the supplier which has the minimum ordering cost, i.e., $j_i = \min_{j \in S}\{c_o(j) \cdot q_i + c_b(j)\}$, so as to align with the objective of minimizing the total cost. QL4LS is the state-of-the-art reinforcement learning based algorithm for the perishable lot-sizing problem under the single-supplier setting. In QL4LS, a state is defined as a pair of two real numbers, one equal to the inventory level of the materials and the other equal to the sum of the remaining life times of materials in the inventory, and an action is a real number representing the order quantity, say q_i. To adapt QL4LS to our problem, we choose the supplier which generates the least ordering costs, i.e., $j_i = \min_{j \in S}\{c_o(j) \cdot q_i + c_b(j)\}$ (s.t. the problem becomes one under the single-supplier scenario), apply the QL4LS algorithm based on the chosen supplier, and set the order quantity from the chosen supplier to q_i in each time period.

Hyperparameter Selection and Model Training. For the baseline algorithms, all hyperparameters are set to be the same as those in existing studies. For all algorithms, we regard each material as an independent agent for each problem instance of size $|M| \times |S|$. For the dataset, which consists of data spanning over 6 years, from 2015 to 2020, we use the first five years for training and the data of the last year for testing. The network $\mu(s; \theta)$ is composed of one hidden layer and one output layer. We use the tanh function as the activation function with 64 neurons in the hidden layer, and we use sigmoid function with $|S|$ neurons in the output layer corresponding to different actions. The network $Q(s, a; \omega)$ consists of 2 layers. In the first layer, we use the tanh function as the activation function with 64 neurons, and in the second layer, we use a linear function with $|S|$ neurons as the output corresponding to different actions. For training, we set the discount factor γ to 1, and use the Adam optimiser with constant learning rate 10^{-4}. We also adopt $\epsilon = 0.1$ for the ϵ-greedy process involved in training process. Other parameters follow the default settings in PyTorch. We train the networks for 20,000 iterations. The hardware we use is a machine with Intel Core i9-10940X CPU and a single Nvidia GeForce 2080Ti GPU. Our codes can be found via https://github.com/wangkaixin219/RL4LS.

5.2 Experiment Results

(1) Results on Real Dataset. The average time of training a satisfactory model is around 1.4 hours and the learning curve for RL4LS on real dataset is presented in Figure 2(a).



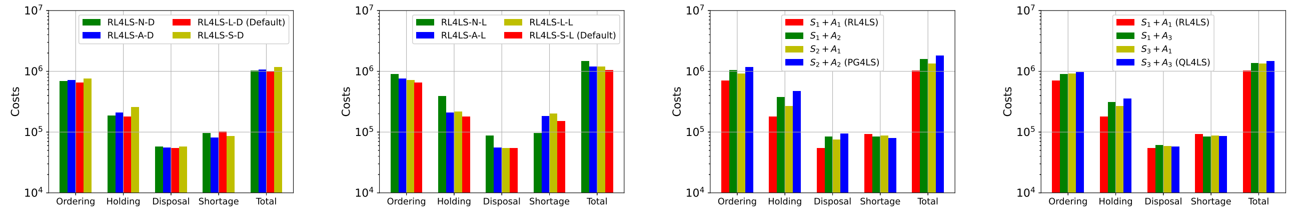
(a) Learning Curve (b) Cost Decomposition

Fig. 2. Results on real dataset.

We can see that the total cost gradually decreases over the training process, and the algorithm converges after 18,000 episodes. Consider the total costs (Figure 2(b)), RL4LS outperforms all existing algorithms. For all but GA, the costs mainly come from the holding costs and ordering costs, which are almost 80% of the total costs. When we order the materials more than the quantity we really need, it incurs higher purchasing costs and holding costs. For GA, the cost mainly comes from the shortage costs since it orders much fewer materials but they cannot meet the demands. The possible reasons are (1) GA gradually stucks at the local optimum and (2) the unit cost of having a shortage of the material, i.e., c_s, is around 7-9 times of that of purchasing, i.e., $c_k(j)$. Consider the efficiency (Table 2). We observe that QL4LS runs the fastest, which could be explained by that (1) GA and IBFA need a large amount of time for trial-and-error and (2) the state and the action are both cheaper to compute compared with RL4LS and PG4LS. In addition, RL4LS runs the second fast, only slightly slower than QL4LS.

(2) Ablation Study on Prediction Models (Real Dataset). Recall that in Section 4.1, we make the predictions on both the lead time and the demands in our state definition. We choose three models, namely ARIMA [38], LSTM [39] and Gaussian Sampling. For ARIMA(p, d, q) prediction method, we set $p = 1, d = 0, q = 1$ for both the demand and the lead time predictions. For the demand prediction (resp. lead time prediction of a supplier) using LSTM, the input of LSTM is a 12-dimensional (resp. 3-dimensional) vector, which consists of the demand history data of the last 12 periods (resp. the last 3 lead time history data of the same supplier). For Gaussian Sampling, we assume that the demand and lead time are both following some Gaussian distributions. We denote these variants by RL4LS-$N/A/L/S$-D/L, where N represents we do not use prediction, A (L or S) represents the prediction model is using ARIMA (LSTM or Sampling), and D (L) represents the model is for the demand (lead time) prediction. Consider the costs (Figure 3(a) and Figure 3(b)). For the demand prediction, using LSTM performs the best among these four variants, which could be explained by the fact that using LSTM can make a more accurate prediction than other methods, and this information could be used to decide the actions with the least costs. For lead time prediction, however, using sampling method outperforms the others. We observe that LSTM and ARIMA do not work for lead time prediction. The reason is that the lead time is always affected by some factors that unpredictable, such as weather or traffic, which makes the prediction unreliable. Consider the running time (Table 3). We observe that all algorithms are efficient (i.e. the running time $<$ 3s).

(3) Ablation Study on State and Action Definitions (Real Dataset). To evaluate the effectiveness of the state and action definitions of RL4LS, separately and collectively, we replace each of them with some alternative definitions,



(a) Prediction on demands (b) Prediction on lead time (c) Combination with S_2 and A_2 (d) Combination with S_3 and A_3

Fig. 3. Results on ablation study.

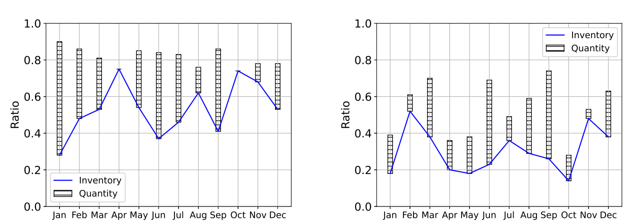
TABLE 3
Running time (seconds) on ablation study (demand and lead time).

Alg.	RL4LS-N-D	RL4LS-A-D	RL4LS-L-D	RL4LS-S-D
Time	2.11	2.26	2.39	2.18
Alg.	RL4LS-N-L	RL4LS-A-L	RL4LS-L-L	RL4LS-S-L
Time	1.98	2.57	2.74	2.39

TABLE 4
Running time (seconds) on ablation study (state and action).

Alg.	$S_1 + A_1$	$S_1 + A_2$	$S_2 + A_1$	$S_2 + A_2$
Time	2.39	2.24	2.90	2.77
Alg.	$S_1 + A_1$	$S_1 + A_3$	$S_3 + A_1$	$S_3 + A_3$
Time	2.39	2.26	1.95	1.78

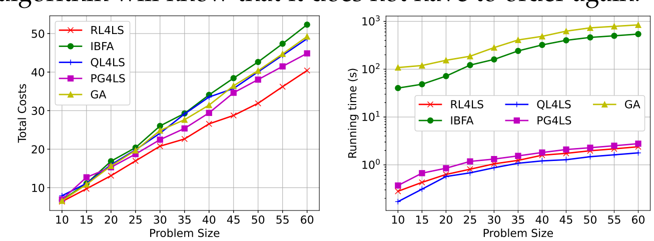
namely those adopted in PG4LS [4] and QL4LS [6]. We denote the state and action definitions of RL4LS (PG4LS, QL4LS) by S_1 and A_1 (S_2 and A_2, S_3 and A_3). In this experiment, we explore two groups of combinations. In group one, we explore (1) $S_1 + A_1$ (i.e., RL4LS), (2) $S_1 + A_2$, (3) $S_2 + A_1$, and (4) $S_2 + A_2$ (i.e., PG4LS). In group two, we explore (1) $S_1 + A_1$ (i.e., RL4LS), (2) $S_1 + A_3$, (3) $S_3 + A_1$, and (4) $S_3 + A_3$ (i.e., QL4LS). Consider the effectiveness (Figure 3(c) and Figure 3(d)). We observe that $S_1 + A_1$ (i.e., RL4LS) performs the best in both groups while $S_2 + A_2$ (i.e., PG4LS) and $S_3 + A_3$ (i.e., QL4LS) perform the worst in each group, respectively. For both groups, RL4LS has the least ordering and holding costs, which dominate the total costs. This could be possibly explained by that a state of RL4LS captures richer and more relevant information of the inventory and the action orders the materials timely and adequately. Consider the efficiency (Table 4). For group one, we observe that $S_1 + A_2$ runs the fastest, which could be explained by that (1) S_1 is cheaper to compute than S_2 and (2) compared with A_2, A_1 needs to decide the quantity and choose the best supplier, which will increase computation load. For group two, we observe that RL4LS is not so efficient which could be explained that S_3 and A_3 are both cheaper to compute than S_1 and A_1.



(a) Case study for Material-I (b) Case study for Material-II

Fig. 4. Results on real dataset (case study).

(4) Case Study (Real Dataset). We choose two typical materials. Material-I has an extremely high unit shortage cost c_{s_1} but its disposal cost c_{d_1} is low while Material-II does not cost too much when shortage happens, i.e., c_{s_2} is much smaller than c_{s_1}, but discarding will result in a large punishment c_{d_2}. These two materials have the same fixed lifetime. The results are presented in Figure 4. The blue lines of these two figures represent the inventory change while the bars represent the order quantity on the corresponding state. For Material-I, maintaining a high inventory level is a good choice since low inventory level sometimes may encounter the problem of demand unmet. Thus, in Figure 4(a), we can see that the inventory level will almost maintain at a level over 80% of the maximum capacity. The policy for Material-II tells another story in Figure 4(b). Since discarding means a large punishment, the order quantity and the inventory level maintain at a low level. From the result we also observe another interesting phenomenon. From July to October, the inventory level decreases gradually. Instead of ordering a large amount of materials at October when the inventory level is the lowest, it orders the least at that time period. The explanation is that we have ordered those materials in August and September in advance. Since we have encoded the information of in-transit material in our state vector, the algorithm will know that it does not have to order again.



(a) Total costs ($\times 10^7$) (b) Running time

Fig. 5. Results on synthetic dataset.

(5) Results on Synthetic Dataset. The results of the synthetic datasets are present in Figure 5. Since the demands and lead time generated in [8] follow a uniform distribution, we use the same distribution to sample the demand and lead time for the predictions in the state definition. Consider the effectiveness, our proposed algorithm outperforms other algorithms under most cases. GA has the second best results when the problem size is small (i.e., problem size $< 20 \times 20$) since GA could generate a large search space due to the chromosome structure, and it is easy to extract a good solution. However, when the problem size increases, it becomes harder to search the whole solution space, and as a result, the performance becomes worse than the RL-based algorithms. Compared with other RL-based algorithms, our algorithm is also competitive since the our proposed state definition captures more information than that of QL4LS and PG4LS. As for the running time, GA and IBFA need a large amount of time by trial-and-error while RL-based algorithms are efficient as the problem size becomes larger. QL4LS runs the fastest, which could be explained by the fact that the state is cheaper to compute.

(6-7) Results on other parameter studies (zero lead time, constant demand and varying the maximum time period). They can be found in the technical report [1].

6 Conclusion

In this paper, we study a novel problem, namely LS-PMU, where we aim to minimise the total costs for perishable materials under various uncertainties. We propose a reinforcement learning based algorithm, i.e., RL4LS. Compared with existing algorithm, our algorithm can intelligently choose a supplier and decide an order quantity so as to minimise the overall costs. Extensive experiments on real and synthetic datasets demonstrate that our algorithm is effective and efficient. It is worthy of noting that our proposed solution is not suitable for the materials whose associated costs vary significantly in different time periods since those costs are considered consistent in our problem. Also, since we do not consider a fill rate (fr), i.e., the ratio of the satisfied demands to the total demands, in our model. If the inventory has such a goal, e.g., $fr \geq 95\%$, our solution would not guarantee to achieve it.

References

[1] K. Wang, C. Long, D. J. Ong, J. Zhang, and X. Yuan, "Single-site perishable inventory management under uncertainties: A deep reinforcement learning approach (technical report)," https://wangkaixin219.github.io/usage/inventoryTR.pdf.

[2] E. Ahmadi, D. T. Masel, S. Hostetler, R. Maihami, and I. Ghalehkhondabi, "A centralized stochastic inventory control model for perishable products considering age-dependent purchase price and lead time," *Top*, vol. 28, no. 1, pp. 231–269, 2020.

[3] A. K. Sinha and A. Anand, "Optimizing supply chain network for perishable products using improved bacteria foraging algorithm," *Applied Soft Computing*, vol. 86, p. 105921, 2020.

[4] Z. Peng, Y. Zhang, Y. Feng, T. Zhang, Z. Wu, and H. Su, "Deep reinforcement learning approach for capacitated supply chain optimization under demand uncertainty," in *2019 Chinese Automation Congress (CAC)*. IEEE, 2019, pp. 3512–3517.

[5] L. Kemmer, H. von Kleist, D. de Rochebouët, N. Tziortziotis, and J. Read, "Reinforcement learning for supply chain optimization," in *European Workshop on Reinforcement Learning 14*, 2018, pp. 1–9.

[6] A. Kara and I. Dogan, "Reinforcement learning approaches for specifying ordering policies of perishable inventory systems," *Expert Systems with Applications*, vol. 91, pp. 150–158, 2018.

[7] A. Oroojlooyjadid, M. Nazari, L. Snyder, and M. Takáč, "A deep q-network for the beer game: A deep reinforcement learning algorithm to solve inventory optimization problems," *arXiv preprint arXiv:1708.05924*, 2017.

[8] H. Soleimani, M. Seyyed-Esfahani, and M. A. Shirazi, "Designing and planning a multi-echelon multi-period multi-product closed-loop supply chain utilizing genetic algorithm," *The International Journal of Advanced Manufacturing Technology*, vol. 68, no. 1-4, pp. 917–931, 2013.

[9] H. M. Wagner and T. M. Whitin, "Dynamic version of the economic lot size model," *Management science*, vol. 5, no. 1, pp. 89–96, 1958.

[10] N. Brahimi, N. Absi, S. Dauzère-Pérès, and A. Nordli, "Single-item dynamic lot-sizing problems: An updated survey," *European Journal of Operational Research*, vol. 263, no. 3, pp. 838–863, 2017.

[11] E. Berk and Ü. Gürler, "Analysis of the (q, r) inventory model for perishables with positive lead times and lost sales," *Operations Research*, vol. 56, no. 5, pp. 1238–1246, 2008.

[12] F. Olsson and P. Tydesjö, "Inventory problems with perishable items: Fixed lifetimes and backlogging," *European Journal of Operational Research*, vol. 202, no. 1, pp. 131–137, 2010.

[13] Y. Pochet and L. A. Wolsey, *Production planning by mixed integer programming*. Springer, 2006, vol. 149, no. 2.

[14] N. Brahimi, S. Dauzere-Peres, N. M. Najid, and A. Nordli, "Single item lot sizing problems," *European Journal of Operational Research*, vol. 168, no. 1, pp. 1–16, 2006.

[15] N. Absi, S. Kedad-Sidhoum, and S. Dauzère-Pérès, "Uncapacitated lot-sizing problem with production time windows, early productions, backlogs and lost sales," *International Journal of Production Research*, vol. 49, no. 9, pp. 2551–2566, 2011.

[16] I. Saracoglu, S. Topaloglu, and T. Keskinturk, "A genetic algorithm approach for multi-product multi-period continuous review inventory models," *Expert Systems with Applications*, vol. 41, no. 18, pp. 8189–8202, 2014.

[17] A. Akbalik, B. Penz, and C. Rapine, "Capacitated lot sizing problems with inventory bounds," *Annals of Operations Research*, vol. 229, no. 1, pp. 1–18, 2015.

[18] S. H. R. Pasandideh, S. T. A. Niaki, and S. M. Mousavi, "Two metaheuristics to solve a multi-item multiperiod inventory control problem under storage constraint and discounts," *The International Journal of Advanced Manufacturing Technology*, vol. 69, no. 5-8, pp. 1671–1684, 2013.

[19] I. Giannoccaro and P. Pontrandolfo, "Inventory management in supply chains: a reinforcement learning approach," *International Journal of Production Economics*, vol. 78, no. 2, pp. 153–161, 2002.

[20] G. J. Van Zyl, "Inventory control for perishable commodities," North Carolina State University. Dept. of Statistics, Tech. Rep., 1963.

[21] S. Nahmias, "The fixed-charge perishable inventory problem," *Operations Research*, vol. 26, no. 3, pp. 464–481, 1978.

[22] Nahmias and Steven, "Higher-order approximations for the perishable-inventory problem," *Operations Research*, vol. 25, no. 4, pp. 630–640, 1977.

[23] B. E. Fries, "Optimal ordering policy for a perishable commodity with fixed lifetime," *Operations research*, vol. 23, no. 1, pp. 46–61, 1975.

[24] H. N. Chiu, "A heuristic (r, t) periodic review perishable inventory model with lead times," *International Journal of Production Economics*, vol. 42, no. 1, pp. 1–15, 1995.

[25] Chiu and H. Neng, "An approximation to the continuous review inventory model with perishable items and lead times," *European Journal of Operational Research*, vol. 87, no. 1, pp. 93–108, 1995.

[26] ——, "A good approximation of the inventory level in a (q, r) perishable inventory system," *RAIRO-Operations Research*, vol. 33, no. 1, pp. 29–45, 1999.

[27] E. Tekin, Ü. Gürler, and E. Berk, "Age-based vs. stock level control policies for a perishable inventory system," *European Journal of Operational Research*, vol. 134, no. 2, pp. 309–329, 2001.

[28] A. Barman and P. De, "A multi-item deteriorating inventory model under stock level-dependent, time-varying, and price-sensitive demand," in *Recent Trends in Applied Mathematics*. Springer, 2021, pp. 1–12.

[29] L. Feng, J. Zhang, and W. Tang, "Dynamic joint pricing and production policy for perishable products," *International Transactions in Operational Research*, vol. 25, no. 6, pp. 2031–2051, 2018.

[30] D. C. Nnadi, "A decision model for the design and operation of inventory programmes in a manufacturing industry," 2017.

[31] F. Jing and X. Chao, "A dynamic lot size model with perishable inventory and stockout," *Omega*, vol. 103, p. 102421, 2021.

[32] M. Önal, "The two-level economic lot sizing problem with perishable items," *Operations Research Letters*, vol. 44, no. 3, pp. 403–408, 2016.

[33] A. Acevedo-Ojeda, I. Contreras, and M. Chen, "Two-level lot-sizing with raw-material perishability and deterioration," *Journal of the Operational Research Society*, vol. 71, no. 3, pp. 417–432, 2020.

[34] N. Arslan, "Lot sizing with perishable items," Ph.D. dissertation, Bilkent Universitesi (Turkey), 2019.

[35] M. L. Puterman, *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, 2014.

[36] J. Xiong, Q. Wang, Z. Yang, P. Sun, L. Han, Y. Zheng, H. Fu, T. Zhang, J. Liu, and H. Liu, "Parametrized deep q-networks learning: Reinforcement learning with discrete-continuous hybrid action space," *arXiv preprint arXiv:1810.06394*, 2018.

[37] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[38] G. E. Box and D. A. Pierce, "Distribution of residual autocorrelations in autoregressive-integrated moving average time series models," *Journal of the American statistical Association*, vol. 65, no. 332, pp. 1509–1526, 1970.

[39] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[40] Alcon. https://www.alcon.com/.