An approximate optimal ordering policy with multiple deliveries and disposals for short-life time products

Koichi NAKADE* and Ryo TERASHIMA*
*Department of Architecture, Civil Engineering and Industrial Management Engineering
Nagoya Institute of Technology
Gokiso-cho, Showa-ku, Nagoya 466-8555, Japan
E-mail: nakade@nitech.ac.jp

Received: 23 October 2019; Revised: 21 January 2020; Accepted: 13 March 2020

Abstract
In a convenience store, new items like lunch boxes and dairy products are delivered several times in one day. In addition, old unsold items must be disposed because of their expiration dates. The order is made once a day, and the retailer has to decide the number of items ordered based on the current remaining items in the store and the demand distribution. This paper considers an optimal ordering problem with short lifetime products, where each order is made once in one day whereas delivery of ordered products and disposal of out-of-date products are twice in one day. The objective is to find the optimal numbers of items which are delivered twice in the next day, which minimize the total expected discounted cost over an infinite horizon. The model is formulated as a Markov decision process. To overcome curse of dimensionality, the appropriate set of basic functions are investigated to approximate the optimal value function of the Markov decision process. The derived approximate optimal ordering policy for the large-size problems is an unbalanced ordering policy under which more items are ordered in one delivery of one day than in another delivery of the same day even when the demand distribution is time-independent. In addition, it is shown through numerical examples that as the probability is larger that an old item is selected by a customer when both old and new items are in the store, the sub-optimal policy is more unbalanced and gains more profit.

Keywords: Order, Inventory, Markov decision process, Linear programming, Basic function

1. Introduction
1.1 Background and organization
In a supermarket or convenience store, products with a short expiration date such as lunch boxes or sandwiches are sold every day. Due to its short due date, many products remaining after expiration are disposed. On the other hand, if the number of ordered products is too small, the product is sold out many times, which leads to less customers’ satisfaction and will decrease the future demand.

In a convenience store, products are delivered to the store multiple times every day, whereas the order is made once a day. By considering this point, the order must be made efficiently to increase the retailer’s profit. How should the retailer order items for multiple deliveries to maximize the retailer’s profit based on the current amounts of short-life items in the store? Is it appropriate for the retailer to equalize the amounts of orders for multiple deliveries?

To answer the problem, the optimization problem to minimize the retailer’s total expected discounted cost over an infinite horizon is considered. This problem is formulated as a Markov decision process (MDP). Compared with simple policies such as a base stock policy or a fixed order policy, the optimal policy derived by MDP can correspond to each situation and is expected to obtain more profits.

In this model, however, as the size of possible orders and the number of deliveries and disposals increase, the numbers of states and actions increase rapidly. Thus, the optimal policy cannot be computed by standard computing algorithms like the policy iterations due to the huge computation and/or the limitation of the memory size of a computer. Thus, an approximation algorithm must be applied. In this paper, the approximate linear programming method is applied to this ordering problem. Basic functions are defined to approximate optimal value functions. To reduce the size of constraints,
initially constraints on a small subset of states and actions is included, and then new pairs of states and actions that will give better actions are added to the set of constraints. The number of pairs added to the subset is also discussed. For the small size of pairs of states and actions, derived sub-optimal policies are compared with optimal policies computed by the linear program. Through these experiments, the good initial set and updating methods are determined.

Then we use this procedure and derive the sub-optimal ordering policies in several examples. The properties of the sub-optimal policies are investigated under the stationary demand distribution. In particular, the relationship between the first and second orders in one day is investigated. In addition, the probability that each arriving customer selects a new item is changed and its effect on the sub-optimal policy and the optimal profit is discussed.

The remainder of this paper is organized as follows. In Section 2, the model is described, and the problem is formulated as a Markov decision process. The computation method for transition probabilities and one-stage cost is investigated. In section 3, the algorithm of approximate dynamic programming with linear programs, which is described in Saure and Puterman (2017) is introduced. In section 4, the appropriate basic functions are investigated through numerical experiments, and the initial subset of the pairs of states and actions and the number of additional pairs are defined properly by numerical experiments for the small size problem. Then the procedure is applied to the large-size problems and the sub-optimal policies are discussed in this model. In Section 5, a conclusion is given.

1.2 Literature

The optimal amounts of orders and price have been discussed for such products in literature. Petruzzi and Dada (1999) consider the optimal stocking and pricing problem in a single period as a newsvendor problem and the problem in multiple periods are discussed by formulating a Markov decision process. Panda, Saha and Basu (2008) assume that items in inventory deteriorate and the demand depends on stock, and the optimal discount prices in two stages are derived and discussed theoretically and numerically. Li, Lim and Rodrigues (2009) discuss a multi-period discounted reward optimization model with perishable products and limited life time, where old products are sold earlier than the new products. Caro and Gallien (2012) consider a pricing model for a fashion retailer, in which the demand highly depends on the trend of fashion. Chen, Pang and Pan (2014) discuss a pricing and inventory problem for perishable products over a finite horizon. They formulate it as a Markov decision process and discuss monotonicity properties of optimal policies. Nakade and Ikeuchi (2016) consider an optimal ordering and pricing problem where new and clearance goods are sold at the same time.

In literature, each order in one day corresponds to the delivery in another day. In a convenience store, however, products are delivered to the store multiple times every day, whereas the order is made once a day. Such multiple deliveries in one day for short-life time items have not been analyzed.

In this paper, to overcome the curse of dimensionality, the approximate linear programming method is applied to this ordering problem (see Schweitzer and Seidman (1985) and Saure and Puterman (2017)). Usually, one pair of states and actions is added to the small-size linear program in each iteration. In our problem with multiple orders and deliveries, however, this procedure is time-consuming because many iterations are needed. To reduce the computation time for deriving the sub-optimal order policy in our problem, we decide the size of the additional pairs added to the linear programs for approximate solutions, as well as a set of basic functions, through preliminary numerical experiments.

2. Model

2.1 Model description

An optimal ordering problem of short lifetime products is considered. There are two delivery epochs and two disposal epochs every day. New products are ordered at the beginning of the day before they reach a retailer. At each of disposal epochs, the products left on the shelf, which are delivered at the first or second delivery epoch one day before, are discarded.

In this model, in the first and third periods, there are two types of products, one is delivered earlier and the other is delivered later. The former is called an old item and the latter is called a new item. When the numbers of old and new items being sold are \(m\) and \(n\) respectively, the state of the store, which is called “the inventory state”, is shown as \((m,n)\). In the second and fourth periods, there are products which are delivered at a certain time. If there are \(n\) items, the inventory state is shown as \((0,n)\).

The retailer orders a single type of products at the beginning of each day. Each order consists of two numbers, one of them is the number of items which will be delivered at the first delivery epoch of the next day, and the other is the
The number of items which will be delivered at the second delivery epoch. The possible maximum number of orders for each delivery epoch is defined as \( Q \).

The ordering, delivery and disposal epochs in one day are sequenced in the following (see also Fig. 1). At the beginning of the day, the numbers of old and new items are \( x_1 \) and \( x_2 \), respectively, and the order is made. The values of \( a_1 \) and \( a_2 \) are the numbers of ordered items which are delivered at the first and second delivery epochs in the next day, respectively. Then after first period of the day, the numbers of old and new items are \( x_11 \) and \( x_21 \), respectively, and old items, which were delivered at the first delivery time yesterday, are thrown away. Then at the end of the second period, new products arrive at the retailer. This epoch is called the first delivery epoch. Then the third period starts, and at its end the remaining products are disposed. Then the fourth period begins. At its end, which is the second delivery epoch and also the end of the day, new products arrive. Just after the arrival, the next day starts and the order is made.

The demand of products during the \( i \)-th period follows a probability distribution \( \{d_l^i; l = 0,1, ..., l_{\text{max}}^i \} \) for \( i = 1,2,3,4 \), where \( l_{\text{max}}^i \) is the maximal number of the demand for the \( i \)-th period. These amounts of demand among different periods are mutually independent. When \( m \) old products and \( n \) new products are sold on the shelf of the retailer at the same time, the arriving customer selects a new item with probability \( P_{m,n} \) and selects an old item with probability \( 1 - P_{m,n} \).

The cost structure is as follows: the profit for selling one item is \( R \), the disposal cost per one item is \( D \), and the lost sale cost per one unsatisfied demand is \( L \). In this problem the life time of items is very short, and many items in inventory lead to the disposal of items directly, and thus the holding cost for items is not included. The objective is to determine the optimal ordering policy which minimizes the total discounted cost over an infinite horizon.

2.2 Nomenclature

Important symbols are defined. Symbols similar to one of the followings are not listed. The variables which appear in the linear programs are defined in section 3.

\( \{d_l^i; l = 0,1, ..., l_{\text{max}}^i \} \): the probability of demand in the \( i \)-th period,

\( m,n \): the inventory state,

\( P_{m,n} \): the probability that arriving customer selects a new item in an inventory state \( (m,n) \),

\( Q \): the maximum of the number of items in each order,

\( R \): the profit for selling one item,

\( D \): the disposal cost per one item,

\( L \): the lost sale cost per an unsatisfied demand,

\( S \): a state space of MDP.
s = (x_1, x_2, u_1, u_2) \in S,
A(s): an action space of state s,
a = (a_1, a_2) \in A(s),
p(s'|s, a): a transition probability from state s to state s' when action a is taken,
c(s, a): one-stage cost when state is s and action a is taken,
\gamma: a discount factor,
V^\pi(s): the total discounted cost over an infinite horizon under policy \pi when the initial state is s,
V^\pi(s): the optimal cost function for initial state s,
P^l_{(x_1, x_2)}(x_{11}, x_{21}) (l): the transition probability from inventory state \( (x_1, x_2) \) to \( (x_{11}, x_{21}) \) in the first period when demand is l in this period,
P^l(x_1, x_2)(x_{11}, x_{21}): the transition probability from inventory state \( (x_1, x_2) \) to \( (x_{11}, x_{21}) \) in the first period,
\{\phi_k(s), k = 0, 1, ..., K\}: a set of basic functions.

2.3 Formulation as a Markov decision process

The model described in section 2.1 can be formulated as a Markov decision process.

The decision of orders is made at the beginning of each day. The state is described as necessary information for the decision at this time, and consists of the inventory state \( (x_1, x_2) \) and the order \( (u_1, u_2) \) which is made one day before. Thus, the state space \( S \) of the Markov decision process is given as follows.

\[ S = \{(x_1, x_2, u_1, u_2); x_1, x_2, u_1, u_2 = 0, 1, 2, ..., Q\}. \]

For state \( s \in S \) the action space \( A(s) \) is the set of possible numbers of orders, and it is given by

\[ A(s) = \{(a_1, a_2); a_1, a_2 = 0, 1, 2, ..., Q\}, s \in S. \]

The transition probability that the next state is \( s' = (x'_1, x'_2, u'_1, u'_2) \) when the current state is \( s = (x_1, x_2, u_1, u_2) \) and action \( a \) is taken is denoted by \( p(s'|s, a) \). The expected cost per day when the state is \( s \in S \) and action \( a = (a_1, a_2) \in A(s) \) is taken is denoted by \( c(s, a) \). Let \( \pi \) denote the policy on orders of products which are determined based on the history regarding the sequence of states and actions. Under policy \( \pi \) the expected total discounted cost over an infinite horizon is defined by

\[ V^\pi(s) = E^\pi \left\{ \sum_{t=0}^{\infty} \gamma^t c(s_t, a_t) | (s_0 = s, a_0) \right\}, \quad (1) \]

where \( (s_t, a_t) \) is a pair of the state and the action at the beginning of time \( t \), and \( \gamma \) is a discount factor satisfying \( 0 < \gamma < 1 \). Let the optimal value function be denoted by \( V^\pi(s) \) which satisfies

\[ V^\pi(s) = \inf_\pi V^\pi(s). \quad (2) \]

From the theory of the Markov decision process with a discounted cost, the optimal policy \( \pi^* \) is a deterministic Markov policy and it satisfies

\[ V^{\pi^*}(s) = V^*(s) = \min_\pi V^\pi(s). \quad (3) \]

The action taken by \( \pi^* \) minimizes the right-hand side of the following optimality equation (see Puterman (1994), for example).

\[ V^*(s) = \min_{a \in A(s)} (c(s, a) + \gamma \sum_{s' \in S} p(s'|s, a)V^*(s')), \quad s \in S. \quad (4) \]

2.4 Transition probabilities and one-stage costs

Here, transition probabilities and a one-stage cost are derived.

To derive the transition probability, we discuss the transition of the inventory state from the beginning of each period to the end of this period. In the first period, the inventory state is \( (x_1, x_2) \) and assume \( l \) customers requiring products are assumed to arrive. For each customer, with probability \( P_{x_1, x_2} \) he/she purchases a new product and with probability \( 1 - P_{x_1, x_2} \) an old one is bought. Then the remaining customers buy products in the same way. Thus,
\( P^1_{(x_1, x_2)(x_{11}, x_{21})}(l) \), which is the probability that the inventory state at the end of the first period becomes \((x_{11}, x_{21})\) when the inventory state at its beginning is \((x_1, x_2)\) and \(l\) customers arrive, satisfies the following recursive equations. For \( l > 0 \),

\[
P^1_{(x_1, x_2)(x_{11}, x_{21})}(l) = P_{x_1, x_2} P^1_{(x_1, x_2-1)(x_{11}, x_{21})}(l-1) + (1 - P_{x_1, x_2}) P^1_{(x_{11}, x_{21})(x_{11}, x_{21})}(l-1) \text{ for } x_1 > 0, x_2 > 0,
\]

\[
P^1_{(x_1, 0)(x_{11}, 0)}(l) = 1 \times P^1_{(x_1-1, 0)(x_{11}, 0)}(l-1) \text{ for } x_1 > 0,
\]

\[
P^1_{(0, x_2)(0, x_{22})}(l) = 1 \times P^1_{(0, x_2-1)(0, x_{22})}(l-1) \text{ for } x_2 > 0, \quad P^1_{(0, 0)(0, 0)}(l) = P^1_{(0, 0)(0, 0)}(l-1), \quad (5)
\]

and \( P^1_{(x_1, x_2)(x_{11}, x_{21})}(0) \) is 1 if \((x_1, x_2) = (x_{11}, x_{21})\) and 0 otherwise. Other transition probabilities which do not appear in the above equations are zero. Then transition probability \( P^1(x_1, x_2)(x_{11}, x_{21}) \) from \((x_1, x_2)\) at the beginning of the first period to \((x_{11}, x_{21})\) at its end is given by

\[
P^1(x_1, x_2)(x_{11}, x_{21}) = \sum_{l=0}^{\hat{d}_1} d_1^l P^1_{(x_1, x_2)(x_{11}, x_{21})}(l). \quad (6)
\]

After the end of the first period \( x_{11} \) old items are discarded, and thus the inventory state becomes \((0, x_{21})\). The transition probability from the beginning of the second period to its end \( P^2(0, x_{21})(0, x_{22}) \) can be obtained easily as

\[
P^2(0, x_{21})(0, x_{22}) = d_{x_{21}-x_{22}}^2 \text{ if } x_{22} = 1, 2, \ldots, x_{21}
\]

and

\[
P^2(0, x_{21})(0, 0) = \sum_{l=x_{21}}^{\hat{d}_2} d_2^l(l). \quad (7)
\]

This implies that the transition probability of the inventory state from the beginning of the first period to the end of the second period is given by

\[
P^{12}(x_1, x_2)(0, x_{22}) = \sum_{x_{11}=0}^{x_1} \sum_{x_{22}=0}^{x_2} P^1(x_1, x_2)(x_{11}, x_{21}) P^2(0, x_{21})(0, x_{22}). \quad (8)
\]

In the same way as Eqs. (5) to (7), the transition probabilities in the third and fourth periods, denoted by \( P^3(x_{22}, x_{32})(x_{23}, x_{33}) \) and \( P^4(0, x_{32})(0, x_{34}) \) respectively, can be derived. Then the transition probability of the inventory state from the beginning of the third period to the end of the fourth period is given by

\[
P^{34}(x_{22}, x_{32})(0, x_{34}) = \sum_{x_{23}=0}^{x_{22}} \sum_{x_{33}=0}^{x_{32}} P^3(x_{22}, x_{32})(x_{23}, x_{33}) P^4(0, x_{32})(0, x_{34}). \quad (9)
\]

When the state of a day is \((x_1, x_2, u_1, u_2)\), since \(u_1\) and \(u_2\) items are delivered to the retailer just after the ends of the second and fourth periods respectively, the transition probability of the Markov decision process is given by

\[
P((x_1', x_2', u_1', u_2') | (x_1, x_2, u_1, u_2), (a_1, a_2)) = \sum_{x_{22}=0}^{x_2} P^{12}(x_1, x_2)(0, x_{22}) P^{34}(x_{22}, u_1)(0, x_{34}), \quad (10)
\]

where \(0 \leq x_1' \leq u_1', x_2' = u_2, u_1' = a_1\) and \(u_2' = a_2\).

The expected one-stage cost is computed as follows. The expected costs in the first and the second periods can be computed as
3. Approximate linear programming

3.1 Linear programming approach for Markov decision processes

Under the total expected discounted cost criterion, the optimal policy can be obtained by solving the following linear program (see Puterman (1994)).

\[
\text{Max } \sum_{s \in S} w(s) v(s) \\
\text{s. t. } v(s) - \gamma \sum_{s' \in S} p(s'|s,a) v(s') \leq c(s,a), \quad a \in A(s), \quad s \in S,
\]

where a group of functions \( \{w(s); s \in S\} \) satisfy \( \sum_{s \in S} w(s) = 1, w(s) > 0 \) for \( s \in S \). The solution \( v(s) \) coincides with the optimal value function \( V^*(s) \).

Its dual problem is given by

\[
\text{Min } \sum_{s \in S} \sum_{a \in A(s)} c(s,a) x(s,a) \\
\text{s. t. } \sum_{a \in A(s)} x(s,a) - \gamma \sum_{s' \in S} \sum_{a' \in A(s')} p(s|s',a') x(s',a') = w(s), \quad s \in S, \\
x(s,a) \geq 0, \quad a \in A(s), \quad s \in S.
\]

By solving this dual problem, the optimal action for state \( s \in S \) is the action \( a^*(s) \) which satisfies \( x(s,a^*(s)) > 0. \)

3.2 Approximate linear programming

Here, we give the approximation linear programming, which is based on Saure and Puterman (2017). This is similar to a column generation approach which appears in a combinatorial optimization problem.

The value function is approximately represented as
\[ V^*(s) = \sum_{k=0}^{K} f_k \phi_k(s). \]  

(16)

Here, \( \{ \phi_k(s), k = 0, 1, \ldots, K \} \) is called a set of basic functions, where \( \phi_0(s) = 1 \). By using these basic functions, the linear program (LP) is represented with decision variables \( \{ f_k : k = 0, 1, \ldots, K \} \) as follows:

\[
\begin{align*}
\text{Max}_{f_0, f_1, \ldots, f_K} & \quad \sum_{s \in S} w(s) \sum_{k=0}^{K} f_k \phi_k(s) \\
\text{s.t.} \quad & \sum_{k=0}^{K} f_k \phi_k(s) - \gamma \sum_{s' \in S} p(s'|s, a) \sum_{k=0}^{K} f_k \phi_k(s') \leq c(s, a), \quad a \in A(s), \quad s \in S.
\end{align*}
\]

(17)

This is called an approximate linear program (ALP). By defining

\[
g_k = \sum_{s \in S} w(s) \phi_k(s), \quad h_0 = 1 - \gamma \quad \text{and} \quad h_k(s, a) = \phi_k(s) - \gamma \sum_{s' \in S} p(s'|s, a) \phi_k(s'),
\]

(18)

the program (ALP) is represented as

\[
\begin{align*}
\text{Max}_{f_0, f_1, \ldots, f_K} & \quad f_0 + \sum_{k=1}^{K} g_k f_k \\
\text{s.t.} \quad & h_0 f_0 + \sum_{k=1}^{K} h_k(s, a) f_k \leq c(s, a), \quad a \in A(s), \quad s \in S.
\end{align*}
\]

(19)

The pairs of states and actions are restricted to \( \chi_n \subseteq S \times A \). This problem is called ALP(n).

\[
\begin{align*}
\text{Max}_{f_0, f_1, \ldots, f_K} & \quad f_0 + \sum_{k=1}^{K} g_k f_k \\
\text{s.t.} \quad & h_0 f_0 + \sum_{k=1}^{K} h_k(s, a) f_k \leq c(s, a), \quad (s, a) \in \chi_n.
\end{align*}
\]

(20)

Its dual problems DALP(n) is given as follows:

\[
\begin{align*}
\text{Min} & \quad \sum_{(s, a) \notin \chi_n} c(s, a) \pi(s, a) \\
\text{s.t.} \quad & h_0 \pi(s, a) = 1, \quad \sum_{(s, a) \notin \chi_n} h_k(s, a) \pi(s, a) = g_k, \quad k = 1, 2, \ldots, K, \\
& \pi(s, a) \geq 0, \quad (s, a) \in \chi_n.
\end{align*}
\]

(21)

Starting with a small subset \( \chi_0 \subseteq S \times A \), by adding new pairs \( (s, a) \notin \chi_n \) violating constraint of ALP(n) into \( \chi_n \) in the \( n \)-th iteration, we derive the approximate values of \( V^*(s) \).

The approximate approach is given as follows.

Step 1. Set \( \chi_0 \) and \( n=0 \).

Step 2. Solve ALP(n). The solutions are denoted by \( (f_0^{(n)}, f_1^{(n)}, \ldots, f_K^{(n)}) \).

Step 3. For all pairs \( (s, a) \notin \chi_n \) calculate

\[
v(s, a) = h_0 f_0^{(n)} + \sum_{k=1}^{K} h_k(s, a) f_k^{(n)} - c(s, a),
\]

(22)
and select several pairs \((s, a) \in \chi_n\) which satisfy \(v(s, a) > 0\). If such \((s, a)\) is not found, go to step 4. Otherwise, add them into \(\chi_n\). The new set of pairs is \(\chi_{n+1}\). \(n \leftarrow n + 1\) and go to Step 2.

Step 4. Set
\[
\hat{\varphi}(s) = \sum_{k=0}^{K} f_k^{(n)} \phi_k(s)
\]
and derive the action satisfying
\[
\alpha^*(s) = \arg \min_{a \in A(s)} \{ c(s, a) + \gamma \sum_{s' \in S} p(s'|s, a) \sum_{k=0}^{K} f_k^{(n)} \phi_k(s') \}, \quad s \in S.
\]

The policy with actions \(\{\alpha^*(s); s \in S\}\) is output.

In the standard procedure, one pair of a state and an action is added to \(\chi_n\). On the other hand, if there are many pairs needed to the appropriate approximation, it may be better to add several pairs to \(\chi_n\) at once, to reduce the number of iterations in this procedure. In the following numerical examples, the selection of basic functions, the initial set \(\chi_0\), and the number of pairs of states and actions to \(\chi_n\) are discussed.

### 4. Numerical experiments

#### 4.1 Assumptions and setting of experiments

In the following, results of the numerical experiments are given. Computations are done in a computer with an Intel Core i7-8700 (3.2GHz) CPU and 24GB RAM. Intel C compiler 16 Update 3 is used, and during the execution of the program, linear programs are solved repeatedly with software FICO XPRESS BCL ver 8.5.

To evaluate the validity of the selection of initial subset \(\chi_0\) of states and actions, we first consider the small problems with \(i_{\text{max}}^i = 3\) \((i = 1, 2, 3, 4)\). The demand in each period follows a binomial distribution with parameters \((i_{\text{max}}^i, \alpha)\) where \(i_{\text{max}}^i = 3\) and \(\alpha = 0.3\). \(\gamma\) is set as 0.95, a possible maximal order \(Q\) is 3, and cost parameters are \(R = 200, L = 150\) and \(D = 100\). The probability that an arrival selects a new item in the inventory state \((m,n)\) is \(P_{mn} = \frac{n}{m+n}\).

Optimal actions for several states obtained by solving linear programs are shown in Table 1. The optimal action and the transition from state \((0,0,0,0)\) is drawn in Fig. 2. When the initial state is \((0,0,0,0)\), the optimal action is \((a_1, a_2) = (3,2)\) and the next state is \((0,0,3,2)\). The optimal action in this state is \((2,3)\). The possible next states from \((0,0,3,2)\) are \((0,2,2,3), (1,2,2,3), (2,2,2,3)\) and \((3,2,2,3)\). In this problem, the optimal policy shows that from any state, after a few transitions, the system reaches to one state of \(S^* = \{(0,3,2,2), (1,3,2,2), (2,3,2,2)\}\), whose optimal action is \((2,3)\), and then the state moves to one of \(\{(0,2,2,3), (1,2,2,3), (2,2,2,3)\}\), in which the optimal action is \((2,2)\). Then the system returns to one of states in \(S^*\). In addition, the optimal action for each state is determined only through \((u_1, u_2)\) in a steady state in this case. This implies that the optimal action will highly depend on the previous action.

To develop the appropriate selection of basic functions, we consider three cases:

1. \(\phi_0(s) = 1, \phi_1(s) = x_1, \phi_2(s) = x_2, \phi_3(s) = u_2, \phi_4(s) = u_2\),
2. \(\phi_0(s) = 1, \phi_1(s) = x_1, \phi_2(s) = x_2, \phi_3(s) = u_1, \phi_4(s) = u_2, \phi_5(s) = x_1 x_2, \phi_6(s) = x_1 u_2, \phi_7(s) = x_1 u_2, \phi_8(s) = x_2 u_1, \phi_9(s) = x_2 u_2, \phi_{10}(s) = u_1 u_2\)

The optimal value functions and actions are given in Table 1.

![Fig. 2 Transition under the optimal policy.](image-url)
(iii) In addition to the set of (ii), $\phi_{11}(s) = x_1^2$, $\phi_{12}(s) = x_2^2$, $\phi_{13}(s) = u_1^2$, $\phi_{14}(s) = u_2^2$ are included. When all sets of states and actions are included as $\chi_{0^*}$ in case (i) $f_2$, $f_3$ and $f_4$ are zero, and the obtained actions for any state $(0,0,i,j)$ $(i,j=0,1,2,3)$ are $(0,0)$. Thus, the result is poor. For example, in state $(0,0,0,0)$, many orders are desirable. In addition, value functions estimated by Eq. (23) for these states are the same and the value is -9015.72, whereas under the same policy $\pi$ as derived actions, the values of $V^\pi(s)$ calculated by the following equations are in the range between -6200 to -7000.

$$V^\pi(s) = c(s,a) + \gamma \sum_{s' \in S} p(s'|s,\pi(s))V^\pi(s'), \quad s \in S.$$  \hfill (25)

Thus, the selection (i) results in poor performance. In case (ii) several $f_0, f_9, f_{0*}, f_9$ and $f_{10}$ are zero and the optimal actions for states $(0,0,i,j)$ are $(0,0)$ $(i,j=0,1,2,3)$, and the estimated values of value functions for these states are about -8925. Under the same policy exact value functions are in a range of -5600 to 7200, and thus this set of basic functions do not give the good results.

When case (iii) is applied, values of parameters $f_k$ are shown in Table 2(a), and Table 2(b) shows the estimated values $\hat{V}(s)$ of value functions for the several states $(0,0,i,j)$, which is determined by the minimization of the right hand side of Eq. (24). The exact value $V^\pi(s)$ under the same policy $\pi$ is also presented, which is computed by Eq. (25).

As shown in Table 2, all values of parameters are not zero, and the value functions are approximated to the exact value. In addition, the exact value function $V^\pi(s)$ under the policy is very near to the values of the optimal value function shown in Table 1. The derived actions are $(2,3)$, $(3,2)$ or $(2,2)$ for states $(0,0,i,j)$. The quadratic functions such as $x_1^2$ are needed as basic functions because less inventory leads to more loss of customers, whereas more inventory leads to more disposals. In addition, the pairs of adjacent parameters in the state, like $x_1x_2$, affect on the one-stage cost. In the following, the set of functions (iii) is assumed.

When the subset consisting of pairs of several states and all possible actions are selected as an initial subset $\chi_{0^*}$ the problem cannot be solved by ALP. On the other hand, when an action is fixed and all the pairs of this action and states are selected as an initial subset, the ALP algorithm converges to good solutions. The new pair of state and action is included to $\chi_n$ one by one in step 2 of every iteration. The results when actions $(2,2)$, $(2,3)$ and $(3,2)$ are selected as an initial action are given in Table 3. For each experiment, computation time is about 9 seconds. In this table, in addition to derived actions in several states, values $V^\pi(s)$ computed by solving Eq. (25) under the obtained policy are shown.

Actions derived by ALP are different from optimal actions in several states, but the obtained value of $V^\pi(s)$ is near to the optimal value function $V^*(s)$. In fact, action $(a_1, a_2)$, where $a_1 + a_2$ is a little greater than the expected demand, is expected to be desirable for many states, because an insufficient order quantity will incur a loss of demand and reduces the profit. In addition, in the preliminary experiment, when action $(3,2)$ is selected as an initial action, derived value functions for states shown in this table are the same as optimal value function $V^*(s)$ except $(0,0,0,0)$. On the other hand, when action $(2,3)$ is selected, the derived value functions are different from $V^*(s)$ in several states. Thus $\chi_0$ is set as the set consisting of pairs of all states and one action $(a_1, a_2)$ which satisfies $a_1 + a_2 = 1 + \lceil$the expected amount of demand in one day$\rceil$, where $\lceil x \rceil$ denotes the ceiling of $x$, and if it is odd $a_2 = a_1 + 1$, and otherwise, $a_2 = a_1$.

| (a) Obtained parameters | (b) Approximate values, actions and exact values under the policy |
|-------------------------|------------------------------------------|
| $f_0$                   | $-7921.03$                               |
| $f_1$                   | $-31.54$                                 |
| $f_2$                   | $-268.21$                                |
| $f_3$                   | $-332.00$                                |
| $f_4$                   | $-143.33$                                |
| $f_5$                   | $40.02$                                  |
| $f_6$                   | $8.26$                                   |
| $f_7$                   | $0.54$                                   |
| $f_8$                   | $35.44$                                  |
| $f_9$                   | $3.03$                                   |
| $f_{10}$                | $17.75$                                  |
| $l_0$                   | $0.50$                                   |
| $l_1$                   | $0.02$                                   |
| $l_2$                   | $0.75$                                   |
| $l_3$                   | $0.25$                                   |
| $l_4$                   | $0.45$                                   |
| $l_5$                   | $0.35$                                   |
| $l_{10}$                | $0.50$                                   |
| $\pi_{approx}V^\pi[s]$ | $-7921.03$ (3,2) -7535.42 (3,2) -8394.89 (3,2) -8026.31 |
| $\pi_{exact}V^\pi[s]$  | $-7921.03$ (3,2) -7535.42 (3,2) -8394.89 (3,2) -8026.31 |
| $\pi_{approx}V^\pi[s]$ | $-8044.08$ (3,2) -7777.02 (3,2) -8482.45 (3,2) -8206.64 |
| $\pi_{exact}V^\pi[s]$  | $-8044.08$ (3,2) -7777.02 (3,2) -8482.45 (3,2) -8206.64 |
| $\pi_{approx}V^\pi[s]$ | $-8126.57$ (2,3) -7911.17 (2,3) -8529.45 (2,3) -8306.63 |
| $\pi_{exact}V^\pi[s]$  | $-8126.57$ (2,3) -7911.17 (2,3) -8529.45 (2,3) -8306.63 |
| $\pi_{approx}V^\pi[s]$ | $-8168.51$ (2,2) -7933.49 (2,2) -8535.89 (2,2) -8310.05 |
| $\pi_{exact}V^\pi[s]$  | $-8168.51$ (2,2) -7933.49 (2,2) -8535.89 (2,2) -8310.05 |
| $\pi_{approx}V^\pi[s]$ | $-8205.50$ (3,2) -7816.40 (3,2) -8489.21 (3,2) -8117.28 |
| $\pi_{exact}V^\pi[s]$  | $-8205.50$ (3,2) -7816.40 (3,2) -8489.21 (3,2) -8117.28 |
| $\pi_{approx}V^\pi[s]$ | $-8310.80$ (3,2) -8039.33 (3,2) -8559.02 (3,2) -8245.89 |
| $\pi_{exact}V^\pi[s]$  | $-8310.80$ (3,2) -8039.33 (3,2) -8559.02 (3,2) -8245.89 |
| $\pi_{approx}V^\pi[s]$ | $-8375.55$ (2,2) -8167.51 (2,2) -8310.97 (2,2) -8296.40 |
| $\pi_{exact}V^\pi[s]$  | $-8375.55$ (2,2) -8167.51 (2,2) -8310.97 (2,2) -8296.40 |
In addition, to reduce the computation times, as new pairs that are included in step 3, pairs \((s, a) \notin \chi_n\) which take the greater positive values of

\[ h_0 f_0^{(n)} + \sum_{k=1}^K h_k(s, a) f_k^{(n)} - c(s, a) \]  

are selected.

Table 4 shows the number of iterations, computation time, derived optimal actions for several states when the numbers of added pairs are 1,100,200,256,300. The number of states in this case is 256. When one pair is added to the subset \(\chi_n\), many iterations and computation times are needed. In this case, the obtained policy is sub-optimal. The number of added pairs increases, the computation time decreases. When the number of added pairs is 300, the obtained action is different to the others. In fact, the approximate value functions and actions are the same when the added number of states is 1,100,200,256, whereas when the added number is 300, the approximate value functions and actions are different for several states. In fact, the values of value functions computed by Eq. (25) using the obtained policy in the case of 300 added pairs are greater than those in the case of 256 or less added pairs. In the following examples, the number of selected pairs in each iteration is set to the number of states.

When \(l_{\text{max}}^i\) and \(Q\) are set as 6, the number of states is \(7^4 = 2401\) and the number of pairs of states and actions is \(7^6 = 117649\). Computation time to derive optimal solutions by LP is 190.99 sec., whereas ALP outputs the result with 7 iterations and its total computation time is 2.337 sec., which is less than 1.3 percent of the computation time by LP. Results are given in Table 5. The obtained results by ALP is almost the same as those by LP, and values of the value function by ALP have the same as or very small differences from those by LP.

### Table 3 Computational results of optimal and sub-optimal policies \((Q = l_{\text{max}}^i = 3)\).

| state | Optimal Policy | ALP (initial (2,2)) | ALP(initial (2,3)) | ALP(initial (3,2)) |
|-------|----------------|---------------------|--------------------|--------------------|
| \((0,0,0,0)\) | \((3,2)\) -7539.41 | \((3,2)\) -7511.18 | \((3,2)\) -7523.94 | \((3,2)\) -7525.61 |
| \((0,0,3,2)\) | \((2,3)\) -8315.17 | \((2,3)\) -8285.45 | \((2,3)\) -8308.36 | \((2,3)\) -8315.17 |
| \((0,2,2,3)\) | \((2,2)\) -8710.74 | \((2,2)\) -8685.83 | \((2,2)\) -8704.14 | \((2,2)\) -8710.74 |
| \((0,3,2,2)\) | \((2,3)\) -8734.47 | \((2,3)\) -8708.31 | \((2,3)\) -8727.53 | \((2,3)\) -8734.47 |
| \((1,2,2,3)\) | \((2,2)\) -8635.95 | \((2,2)\) -8611.03 | \((2,2)\) -8629.35 | \((2,2)\) -8635.95 |
| \((1,3,2,2)\) | \((2,3)\) -8616.28 | \((2,3)\) -8590.01 | \((2,3)\) -8609.33 | \((2,3)\) -8616.28 |
| \((2,2,2,3)\) | \((2,2)\) -8516.62 | \((2,2)\) -8476.54 | \((2,2)\) -8510.02 | \((2,2)\) -8516.62 |
| \((2,3,2,2)\) | \((2,3)\) -8482.81 | \((2,3)\) -8456.48 | \((2,3)\) -8475.87 | \((2,3)\) -8482.81 |
| \((3,2,2,3)\) | \((2,2)\) -8383.54 | \((2,2)\) -8343.46 | \((2,2)\) -8376.95 | \((2,2)\) -8383.54 |

### Table 4 The number of iterations, computation time and actions under several numbers of added pairs of states and actions.

| added pairs | 1 | 100 | 200 | 256 | 300 |
|-------------|---|-----|-----|-----|-----|
| iterations | 726 | 8 | 4 | 4 | 4 |
| comp. time(s) | 9.658 | 0.262 | 0.207 | 0.205 | 0.202 |
| \((0,0,0,0)\) | \((3,3)\) | \((3,3)\) | \((3,3)\) | \((3,3)\) |
| \((0,0,3,2)\) | \((2,3)\) | \((2,3)\) | \((2,3)\) | \((2,3)\) |
| \((0,2,2,3)\) | \((2,2)\) | \((2,2)\) | \((2,2)\) | \((2,2)\) |
| \((0,3,2,2)\) | \((2,3)\) | \((2,3)\) | \((2,3)\) | \((2,3)\) |
| \((1,2,2,3)\) | \((2,2)\) | \((2,2)\) | \((2,2)\) | \((2,2)\) |
| \((1,3,2,2)\) | \((2,3)\) | \((2,3)\) | \((2,3)\) | \((2,3)\) |
| \((2,2,2,3)\) | \((2,2)\) | \((2,2)\) | \((2,2)\) | \((2,2)\) |
| \((2,3,2,2)\) | \((2,3)\) | \((2,3)\) | \((2,3)\) | \((2,3)\) |
| \((3,2,2,3)\) | \((2,2)\) | \((2,2)\) | \((2,2)\) | \((2,2)\) |

### Table 5 Computational results of optimal and sub-optimal policies \((Q = l_{\text{max}}^i = 6)\).

| state | Optimal Policy | ALP |
|-------|----------------|-----|
| \((0,0,0,0)\) | \((5,5)\) -19019.05 | \((5,5)\) -18944.20 |
| \((0,0,5,5)\) | \((4,5)\) -20777.95 | \((4,5)\) -20777.95 |
| \((0,5,5,5)\) | \((4,5)\) -21718.52 | \((4,5)\) -21718.52 |
| \((1,5,4,5)\) | \((4,5)\) -21621.23 | \((4,5)\) -21621.23 |
| \((2,5,4,5)\) | \((4,5)\) -21504.75 | \((4,5)\) -21504.75 |
| \((3,5,4,5)\) | \((4,5)\) -21378.17 | \((4,5)\) -21378.17 |
| \((4,5,4,5)\) | \((4,5)\) -21245.59 | \((4,5)\) -21245.59 |
4.2 Sub-optimal ordering in the large-size problems

Here we discuss the large-size problem. \( \mu_{\text{max}} \) and \( Q \) are set as 11, the number of states is \( 12^4 = 20736 \) and the number of pairs of states and actions is \( 12^5 = 2985984 \). The values of derived solutions \( f_k \) are shown in Table 6. The computation time is 187.05 seconds, and the number of iterations is 17. The number of pairs \((s,a)\) included in \( \chi_n \) at the end of computation is 347551.

From Table 6, \( f_1 \) is positive, which implies that the small value of \( x_1 \) is desirable in this problem. The old item in the first period is disposed at its end if it is left in the store. In addition, \( f_3 \), the coefficient of basic function \( u_1 \), is negative, which implies the large value of \( u_1 \) is good, whereas \( f_{13} \), the coefficient of \( u_1^2 \), is positive and greater than that of squares of the other components of the state. This implies that too large \( u_1 \) is not desirable. In fact, the sub-optimal action is \((6,10)\) in the recurrent states under the derived sub-optimal policy, and the recurrent states are \((x,10,6,10)\) where \( x=0,1,\ldots,6 \) (see Fig. 3). This figure shows that the action \((9,11)\) is taken in state \((0,0,0,0)\) and the next state is \((0,0,9,11)\), and then the next action is \((6,10)\) and the next state is \((x_1,11,6,10)\) where \( x_1 \) is from 0 to 9. Then for any state action is \((6,10)\) and the state moves to \((x_1,10,6,10)\) where \( 0 \leq x_1 \leq 6 \). Afterwards, action \((6,10)\) is taken. We note that for any initial state, after several steps the system reaches the same set of states. Thus, the number of ordered items for the first delivery is much smaller than that for the second delivery. The reason is discussed in the following.

The expected amount of demand for each period is 3.3, and the total expected demand in one day is 13.2. Under the obtained policy, at the beginning of the first period of one day, a very few old items and 10 new items are in the store. Then after the first period the number of remaining old items is zero or very small, which are disposed. In the second period, the remaining new items are consumed, but after the second period the probability that several items remain is high. In the beginning of the third period 6 new items are delivered, and after this period a few old items may be disposed. After the fourth period, most of delivered items are consumed because only new items are sold in this period, and at the end of the day items are sold out or there are a very few items because the expected number of demand in these two periods is 6.6. That is, most of items arriving at the first delivery time are consumed only during third and fourth periods, and with a large probability only the items arriving at the second delivery time (which is the end of the day) are consumed in the first and second periods of the next day. The small number of items remain at the beginning of the third period, and new items of the first delivery come to the store. The system repeats this process, and as a result, amounts of both disposal items and lost sales are small.

Table 6 Results of sub-optimal solutions \((Q = \mu_{\text{max}} = 11)\).

| \( f[0] \) | -41484.41 | \( f[5] \) | 5.68 | \( f[10] \) | 3.96 |
| \( f[1] \) | 28.44 | \( f[6] \) | 1.32 | \( f[11] \) | 2.83 |
| \( f[2] \) | -275.25 | \( f[7] \) | 0.31 | \( f[12] \) | 8.59 |
| \( f[3] \) | -398.85 | \( f[8] \) | 11.60 | \( f[13] \) | 18.87 |
| \( f[4] \) | -163.30 | \( f[9] \) | 1.89 | \( f[14] \) | 5.74 |

Fig. 3 Sub-optimal actions for the large-size problem \((P_{m,n} = \frac{n}{m+n})\).
The total expected discounted cost when a recurrent state (3,10,6,10) is an initial state under the obtained policy is -44402.30. On the other hand, under the policy such that the action (8,8) is taken in any state, the total expected discounted cost for the initial state (0,8,8,8) is -43747.12, which is the minimal total cost (the maximal profit) among the cases in which one of the recurrent states is selected as the initial state. When the same numbers of items are ordered for these deliveries, several old items are disposed at the end of both the first and third periods, and more disposal costs occur. Thus, unbalanced orders give less costs (more profits) even when the demand distribution for all periods are the same.

The selection probability \( P_{m,n} \) is changed to the next two cases: (i) \( P_{m,n} = \frac{3n}{m+3n} \) and (ii) \( P_{m,n} = \frac{n}{3m+n} \). In the first case, the probability that a new item is selected is high. Then the derived sub-optimal action is (7,8) or (7,9) in recurrent states (see Fig. 4). In this case, it is desirable for items to be sold out in the first two periods after their delivery. Thus, a balanced and smaller order quantity is preferred. In case (ii) the older one can be selected with large probability, for example by placing the old items on the near side of shelf to customers. Then the suboptimal action is either (6,10) or (6,11) in states of the recurrent class (see Fig. 5). The older one is expected to be sold even in the third period after their delivery. Thus, more orders make more profits.

The total expected discounted cost when the obtained sub-optimal policies are applied \( P_{m,n} = \frac{3n}{m+3n} \) is -40680.88, for the initial (and recurrent) state (7,9,7,8), whereas the total cost when \( P_{m,n} = \frac{n}{3m+n} \) is -46956.10 for the initial state (6,11,6,10). In addition, the total discounted expected costs for the initial state (0,0,0,0) under \( P_{m,n} = \frac{n}{m+n}, \frac{3n}{m+3n} \) and \( \frac{n}{3m+n} \) are -39414.05, -36443.00, -41923.11, respectively. When \( P_{m,n} = 0 \) and 1, the actions in the recurrent states under the sub-optimal policies are (6,10) and (8,7), and the total discounted expected costs for the initial state (0,0,0,0) are -44580.54 and -33718.71, respectively. Thus, when the probability that the old item is purchased by a consumer is large when both items exist in the store, more profits are obtained by using the obtained sub-optimal policy. Thus, it is important to sell the old items earlier.

![Fig. 4](image4.png) Sub-optimal actions for the large-size problem \( P_{m,n} = \frac{3n}{m+3n} \).

![Fig. 5](image5.png) Sub-optimal actions for the large-size problem \( P_{m,n} = \frac{n}{3m+n} \).
5. Conclusion

In this paper, the optimal ordering problem for short lifetime products is considered. The problem is formulated as a Markov decision process. To solve the large-size problem with overcoming the curse of dimensionality, the approximate linear programming approach is applied. By setting the basic functions, the initial subset of states and actions and the number of pairs added to the subset appropriately, the sub-optimal policies can be obtained. For the larger-size problem, results obtained by this approach show that an unbalanced action gives the best result, by allocating more items in one delivery time consumed in long periods and less items in the other delivery time consumed in a short time, in particular when the probability that an old item is selected by a customer is large. In addition, as this probability is larger, more profit is gained under the sub-optimal policy.

The problem is generalized to the situation where three or more disposals and deliveries in one day. The state may include the other factor such as climate, temperature and so on. In this case, modification of the computation method may be needed. Comparison of the proposed approximate linear programming with the other approach of approximate dynamic programming is also important. These are left for future research.

References

Caro, F. and Gallien, J., Clearance pricing optimization for a fast-fashion retailer, Operations Research, Vol. 60, No. 6 (2012), pp. 1404-1422.
Chen, X., Pang, Z. and Pan, L., Coordinating inventory control and pricing strategies for perishable products, Operations Research, Vol. 62, No. 2 (2014), pp. 284-300.
Li, Y., Lim, A. and Rodrigues, B., Pricing and inventory control for a perishable product, Manufacturing and Service Operations Management, Vol. 11, No. 3 (2009), pp. 538-542.
Nakade, K. and Ikeuchi, K. Optimal ordering and pricing on clearance goods, International Journal of Industrial Engineering, Vol. 23, No. 3 (2016), pp. 155-165.
Panda, S., Saha, S. and Basu, M., An EOQ model for perishable products with discounted selling price and stock dependent demand, Central European Journal of Operations Research, Vol. 17 (2009), pp. 31-53.
Petruzzi, N. C. and Dada, M., Pricing and the newsvendor problem: a review with extensions, Operations Research, Vol. 47, No. 2 (1999), pp. 183-194.
Puterman, M. L., Markov Decision Processes: Discrete Stochastic Dynamic Programming (1994), Wiley.
Saure, A. and Puterman, M. L., Advance patient appointment scheduling, in R. Boucherie and N.M. van Dijk (eds), Markov Decision Processes in Practice (2017), Chapter 8, pp. 245-268, Springer.
Schweitzer, P. and Seidman, A., Generalized polynomial approximations in Markovian decision processes, Journal of Mathematical Analysis and Applications, Vol. 110, No. 2 (1985), pp. 568-582.