Empirical Performance of Lexicographic Bi-criteria Solutions during an Operating Run of Repetitive Packaging by a Combinatorial Weigher

Yoshiyuki Karuno\textsuperscript{a} and Ryo Saito\textsuperscript{b}

\textsuperscript{a}Faculty of Mechanical Engineering, Kyoto Institute of Technology, Kyoto 606-8585, Japan
\textsuperscript{b}Graduate School of Science and Technology, Kyoto Institute of Technology, Kyoto 606-8585, Japan

\* Corresponding Author: karuno@kit.ac.jp

Abstract

In this paper, a lexicographic bi-criteria combinatorial optimization problem arising in actual food packaging equipments and its solutions are considered. In each packaging operation, a set \( I = \{ i \mid i = 1, 2, \ldots, n \} \) of current \( n \) items (for example, \( n \) green peppers) with their weights \( w_i \) and priorities \( p_i \), and a target weight \( t \) of a package are given. Then the problem asks to find a subset \( I' \subseteq I \) so that the total weight \( \sum_{i \in I'} w_i \) is minimized as the primary objective, meeting the target weight constraint \( \sum_{i \in I'} w_i \geq t \), and the total priority \( \sum_{i \in I'} p_i \) is maximized as the second objective. Such a problem is repeatedly solved during an operating run of the food packing system, in which a large number of packages are produced one by one. The priority of an item is defined to be the duration of the item in the food packing system, and it is expected to avoid the frequent occurrence of items with longer durations. In this paper, all input data are assumed to be integral, and the effectiveness of the lexicographic bi-criteria solutions in an operating run is empirically observed by means of an \( O(nt) \) time exact algorithm for the food packing problem.

Keywords: Combinatorial food packaging, Lexicographic bi-criteria modeling, Pseudo-polynomial time algorithms, Repetitive optimization.

1. Introduction

In this paper, we deal with a combinatorial optimization problem arising in food production, which has been introduced to represent an automated packaging operation preformed by actual packaging equipments\textsuperscript{(1,2)}, so-called automatic combination weighers and multi-head weighers\textsuperscript{(3,4)}. The problem is mathematically the same as that treated in previous works by the authors and so on. For example, in our recent work\textsuperscript{(5)}, we have designed a polynomial time heuristic algorithm for the problem in each packaging operation, and we have analyzed the theoretical and numerical performance. On the other hand, in this paper we empirically focus on the repetitive behavior of the exact solutions and consecutive instances of the problem during an operating run, i.e., a series of iterations of the packaging operation.

As shown in Fig. 1, the food packing system is typically modeled as a packaging mechanism consisting of \( n \) weighing hoppers\textsuperscript{(6)}. Some amount of foods (such as a green pepper, a ham, a handful of potato chips, and so on) is thrown into each hopper, and we call it an item. Given a set \( I \) of current \( n \) items in hoppers, the food packing system chooses a subset \( I' \subseteq I \) of items, utilizing their weights and some other information. Then, the chosen items are put into a single package. The resulting empty hoppers are supplied with next new items, and the set \( I \) is updated by taking the union of the remaining items in \( I - I' \) and the new items. The packaging operation is repeated to produce a large number of packages one by one. Some of actual food packaging equipments are able to produce more than two hundred packages per minute.
at the maximum\(^3,4\).

The food packing system usually chooses some items in the current hoppers without knowing the weights of next new items. From this fact, most of mathematical models have been formulated in each packaging operation\(^7\)\(^{-11}\). Probably as the most important issue in each packaging operation, the total weight of chosen items for each package must be no less than a specified target weight from the service conscience. The target weight constraint is a hard constraint of the problem of choosing a subset \(I'\) \(^6\). Let \(w_i\) denote the weight of item \(i\), i.e., the current item in the \(i\)-th hopper, and let \(t\) denote the target weight. Then, the target weight constraint is represented by

\[\sum_{i \in I'} w_i \geq t.\]

The primary objective of each packaging operation is to minimize \(\sum_{i \in I'} w_i\) under the target weight constraint. In other words, this aims at minimizing the amount of surplus in each package.

Further, we sometimes worry about items with longer durations in the current hoppers. That is, an item may be left in hopper for a long time before it is chosen to be packaged during an operating run\(^8\). In order to avoid such a situation, a priority \(p_i\) has been introduced into each item \(i\) \(^{1,2}\), and it is defined by a non-decreasing function of the duration in hopper of the item. Then, the total priority \(\sum_{i \in I'} p_i\) is to be maximized as the second objective of each packaging operation. In this paper, we also expect the items with longer durations in hoppers to be preferably chosen by regarding the second objective.

The problem of choosing a subset \(I'\) from the set \(I\) of current \(n\) items with respect to the total weight and total priority objectives is viewed as a lexicographic bi-criteria 0-1 integer programming problem. The problem even with the total weight objective only is NP-hard, while it admits an \(O(nl)\) time exact algorithm\(^{10}\), i.e., the problem can be solved in pseudo-polynomial time. For the quality with respect to the total weight of a package, some other interesting numerical results have also been reported\(^7,9\). However, the food packing algorithms employed in the numerical experiments basically enumerate \(O(2^n)\) subsets \(I'\) of the set \(I\) from the viewpoint of the \(O\) notation.

Recall that in each packaging operation, after choosing a subset \(I' \subseteq I\) of items for the current problem instance, the other items in \(I - I'\) are left in the hoppers. This means that the input data of the subsequent problem instance for the next packaging operation depend on the chosen subset \(I'\) of current items. In this paper, we therefore conduct numerical experiments to observe another aspect of the repetitive behavior of the exact solutions and consecutive problem instances during an operation run. In order to compute an exact solution for each problem instance generated in an operating run, we utilize the \(O(nlt)\) time exact algorithm mentioned in the above\(^{10}\).

### 2. Preliminaries

In this section, we review a problem formulation of the lexicographic bi-criteria food packing problem in each packaging operation. As in one of our previous works\(^5\), we call the food packing problem with respect to the lexicographic bi-criteria (that is, the total weight of chosen items as the primary objective and total priority of the chosen items as the second objective) LEXICO for short. When we regard only the primary objective of the total weight of chosen items, we call the food packing problem PRIMARY.

Let \(n\) denote the number of weighing hoppers in the food packing system to be treated in this paper (see again Fig. 1), and let \(\ell_{\text{max}}\) denote the total number of packages to be produced during an operating run, which is equivalent to the number of iterations of packaging operation during the operating run.

Unless otherwise stated, item \(i\) means the current item in the \(i\)-th hopper. Let \(\ell\) denote the current iteration number of packaging operation \((1 \leq \ell \leq \ell_{\text{max}})\), and let \(\ell_i\) denote the iteration number at which item \(i\) has been thrown into the \(i\)-th hopper when the hopper was empty. Then, we refer to

\[d_i(\ell) = \ell - \ell_i + 1 \quad (\geq 1)\]

as the duration in hopper of item \(i\). We are going to set the priority of item \(i\) below to be the duration. On the other hand, for notational simplicity in this section, the current iteration number \(\ell\) is dropped from the following notations of an instance of problem LEXICO in each packaging operation. In the next section, the \(\ell\) of the current iteration number will be called back to discuss more than one instance in an operating run at the same time.

An instance of problem LEXICO in each packaging operation consists of the following input data:

- \(I = \{i \mid i = 1, 2, \ldots, n\}\): A set of current \(n\) items.
- \(w_i\): A positive integer weight of each item \(i \in I\).
- \(p_i\) \((:= d_i(\ell))\): A positive integer priority of each item \(i \in I\).
- \(\ell\): A prescribed target weight for each package, which is also assumed to be a positive integer.

For notational convenience, the maximum weight and the sum weight over the \(n\) items in the set \(I\) are denoted by

\[w_{\text{max}} = \max_{1 \leq i \leq n} \{w_i\}\quad \text{and} \quad w_{\text{sum}} = \sum_{i=1}^{n} w_i,\]

respectively.

Problem LEXICO is formulated by means of a 0-1 vector \(x = (x_1, x_2, \ldots, x_n)\), instead of a subset \(I' \subseteq I\) of items,
where

\[ x_i = \begin{cases} 1 & \text{if item } i \text{ is chosen for } I', \\ 0 & \text{otherwise.} \end{cases} \tag{3} \]

**Problem LEXICO**

minimize \( f(x) = \sum_{i=1}^{n} w_i x_i \) as the primary objective, \( \tag{4} \)

maximize \( g(x) = \sum_{i=1}^{n} p_i x_i \) as the second objective, \( \tag{5} \)

subject to \( \sum_{i=1}^{n} w_i x_i \geq t, \tag{6} \)

\[ x_i \in \{0, 1\}, \quad i = 1, 2, \ldots, n. \tag{7} \]

The target weight constraint and binary constraints of variables \( x_i \) are represented by Eqs. (6) and (7), respectively. A solution \( x = (x_1, x_2, \ldots, x_n) \) satisfying Eqs. (6) and (7) is referred to as a feasible solution of problem LEXICO. The primary objective of Eq. (4) aims at attaining the total weight of chosen items in a feasible solution as close to the target weight \( t \) as possible, together with Eq. (6) (i.e., the target weight constraint). The second objective of Eq. (5) is introduced so that the items with longer durations in hoppers are preferably chosen. We assume that

\[ w_{\text{max}} < t \leq w_{\text{sum}}, \tag{8} \]

which implies \( t \leq n \cdot w_{\text{max}} \) (see Eq. (2)).

For a given instance of problem LEXICO, let \( f^* \) denote the minimum of the total weight of chosen items in a feasible solution, and let \( x = \hat{x} \) denote a feasible solution that attains the minimum of the total weight, i.e., \( f^* = f(\hat{x}) \). An optimal solution \( x = x^* \) is defined as a feasible solution such that it satisfies \( f(x^*) = f^* \) and maximizes the total priority among feasible solutions with the minimum total weight \( f^* \), i.e., it satisfies \( g(x^*) \geq g(\hat{x}) \) for any feasible solution \( \hat{x} \) with \( f(\hat{x}) = f^* \). Problem LEXICO asks to find an optimal solution \( x = x^* \). As mentioned before, if we are asked to find a feasible solution \( x = \hat{x} \) with the minimum total weight \( f^* \), we call the problem PRIMARY.

**3. Consecutive Instances in an Operating Run**

In the previous section, we have reviewed the food packing problem in each iteration of packaging operation, while in this section we discuss some features related to the repetitiveness of the packaging operation.

Again, let \( \ell \) denote an iteration number of packaging operation in an operating run (\( 1 \leq \ell \leq \ell_{\text{max}} \)). Let \( I = I^{(\ell)} \) be the \( \ell \)-th set of \( n \) items, and for each item \( i \in I^{(\ell)} \), let \( w_i^{(\ell)} \) and \( p_i^{(\ell)} \) denote the weight and the priority, respectively. Initially (i.e., \( \ell = 1 \)), by Eq. (1), the priority of any item \( i \in I^{(1)} \) is set to be \( p_i^{(1)} := d_i^{(1)} = 1 \).

Suppose that in the \( \ell \)-th iteration of packaging operation, we obtain an optimal solution \( x^{(\ell)} = (x_1^{(\ell)}, x_2^{(\ell)}, \ldots, x_n^{(\ell)}) \), which meets the target weight constraint, of course, i.e., \( \sum_{i \in I^{(\ell)}} w_i^{(\ell)} x_i^{(\ell)} \geq t \). Then, in the subsequent instance with the item set \( I^{(\ell+1)} \), the \( i \)-th item meets

\[ w_i^{(\ell+1)} = w_i^{(\ell)}, \]

\[ p_i^{(\ell+1)} = p_i^{(\ell)} + 1 = d_i^{(\ell)} + 1 \]

if it holds \( x_i^{(\ell)*} = 0 \), since the item \( i \in I^{(\ell)} \) is left in the \( i \)-th hopper of the next \( (\ell + 1) \)-th iteration. Otherwise (i.e., if it holds \( x_i^{(\ell)*} = 1 \)), then the \( i \)-th item in the next \( (\ell + 1) \)-th iteration is a newly supplied item. Since in the iteration number \( \ell_i \) of the new item thrown in the food packing system is obviously \( \ell_i = \ell + 1 \), the priority is set to be

\[ p_i^{(\ell+1)} := d_i^{(\ell+1)} = 1, \]

by Eq. (1). Let

\[ d_{\text{max}}^{(\ell)} = \max_{1 \leq i \leq n} \left\{ d_i^{(\ell)} \right\} = \max_{1 \leq i \leq n} \left\{ p_i^{(\ell)} \right\} \tag{9} \]

and let

\[ d_{\text{mean}}^{(\ell)} = \frac{1}{n} \sum_{i=1}^{n} d_i^{(\ell)} = \frac{1}{n} \sum_{i=1}^{n} p_i^{(\ell)} \tag{10} \]

denote the maximum of the duration of an item and the mean duration over the items, respectively, in the \( \ell \)-th iteration of packaging operation with the item set \( I^{(\ell)} \). Further, we define by

\[ D_{\text{max}} = \max_{1 \leq \ell \leq \ell_{\text{max}}+1} \left\{ d_{\text{max}}^{(\ell)} \right\} \tag{11} \]

the maximum duration among all the items thrown into the food packing system. (We remark that the item set \( I^{(\ell_{\text{max}}+1)} \) is set after the last iteration of packaging operation in the above definition, but of course, the \( (\ell_{\text{max}} + 1) \)-iteration is not performed in the operating run.)

Recall that the food packing system usually chooses some items in the current hoppers without knowing the weights of next new items, and it may be hard to minimize the maximum duration of Eq. (11) directly in some modeling. Therefore, we have considered the total priority as a surrogate for the maximum duration in each packaging operation, i.e., as the second objective in problem LEXICO. We may also be able to define the second objective by the maximum priority of an item among the current \( n \) items in the mathematical modeling, although we do not treat such an objective in this paper. In the following section, we are going to sample the total priorities at some iterations of packaging operation in an operating run by conducting numerical experiments, and discuss the effectiveness of the lexicographic bi-criteria solutions on the improvement of the maximum duration.
4. Numerical Results

As mentioned before, we have already known that an instance of problem LEXICO can be solved by an $O(n t)$ time algorithm\(^{(10)}\), which is a dynamic programming procedure. In this section, we utilize the $O(n t)$ time exact algorithm to solve each of $\ell_{\text{max}}$ instances of problem LEXICO and obtain $\ell_{\text{max}}$ lexicographic bi-criteria solutions in an operating run. We can refer some previous papers for the detail of the $O(n t)$ time exact algorithm\(^{(10,12)}\), and due to the space limitation, we omit the mathematical description of the dynamic programming recursive and the discussion of the correctness in this paper.

In order to observe whether a desirable instance is subsequently delivered by the lexicographic bi-criteria solution to each current instance, we sample some instances among the $\ell_{\text{max}}$ instances in an operating run. Then, for each of the sampling instances, we enumerate all $2^{\ell_{\text{max}}}$ solutions by a brute force algorithm with Gray code\(^{(13)}\). We compute the total weight and the total priority of each of the $2^{\ell_{\text{max}}}$ solutions, even if it is not feasible, and plot each solution as a dot on a so-called scatter diagram.

The instances of problem LEXICO to be tested are randomly generated as follows:

- The number of hoppers: $n = 20$.
- Target weight: $t = 1000$.
- Integer weights: $w_i$’s are uniformly random integers in $[200, 300]$.
- The number of iterations of packaging operation in an operating run: $\ell_{\text{max}} = 10000$.
- The sampling numbers of iterations of packaging operation: $\ell \in \{1, 50, 100, 500, 1000, 5000, 10000\}$.

The program is written in C, and run on a laptop personal computer with Windows 7 Professional (64bit), Intel Core i5 4210M CPU (2.60GHz) and 8GB memory. In order to examine the repetitive behavior of the lexicographic bi-criteria solutions, we also provide the results obtained by solving single criterion instances of problem PRIMARY. Of course, we feed the same sequence of the thrown items into the food packing system for problems LEXICO and PRIMARY both. The number of weighing hoppers in typical food packaging equipments is around twenty\(^{(3,4)}\). The target weight $t = 1000$ illustrates, for example, 100.0 gram in a practical situation.

Figure 2 shows the scatter diagram of the first sampling instance of problem LEXICO, i.e., $\ell = 1$, in (a). The figure also provides the scatter diagram of the first sampling instance of problem PRIMARY in (b). Since the two sampling instances are the same, the scatter diagrams are also the same.

Figure 3 indicates the scatter diagram of the fiftieth sampling instance of problem LEXICO in (a). Also it provides the scatter diagram of the fiftieth sampling instance of problem PRIMARY in (b). The lexicographic bi-criteria solutions from the first to the forty-ninth obtains the maximum duration $d_{\text{max}}^{(50)} = 8$, while the single criterion solutions leave an item in the hopper since the operating run has started, i.e., $d_{\text{max}} = 50$. From this, the scattered area in (a) is narrow, while it in (b) is spread wide in the right space of the frame.

![Fig. 2](image)

**Fig. 2** The plotting of $2^n$ solutions in the first iteration of packaging operation: $d_{\text{max}}^{(1)} = 1$ and $d_{\text{mean}}^{(1)} = 1.0$ in (a) of LEXICO, while $d_{\text{max}}^{(1)} = 1$ and $d_{\text{mean}}^{(1)} = 1.0$ in (b) of PRIMARY

![Fig. 3](image)

**Fig. 3** The plotting of $2^n$ solutions in the fiftieth iteration of packaging operation: $d_{\text{max}}^{(50)} = 8$ and $d_{\text{mean}}^{(50)} = 3.5$ in (a) of LEXICO, while $d_{\text{max}}^{(50)} = 50$ and $d_{\text{mean}}^{(50)} = 15.5$ in (b) of PRIMARY

Figures 4-8 except for Fig. 6 illustrate the similar shapes of scattered areas to that in Fig. 3. In particular, in (b) of Figs. 5 and 8, some of the $2^n$ single criterion solution dots stick out of the frame of the diagram. In the sampling iteration with $\ell = 1000$ (see (b) of Fig. 6), the single criterion solutions before it may coincidentally obtain a smaller maximum duration $d_{\text{max}}^{(1000)} = 33$ than those in the other sampling iterations.

The total weight of a package by the lexicographic bi-criteria solution and the execution time of the $O(n t)$ time exact algorithm on an actual personal computer have been reported by the previous works\(^{(5,10,12)}\), and also due to the space limitation, we omit the detail of them in this paper. Roughly speaking, when $n = 20$ and $t = 1000$, the relative difference of the total weight is less than $(1002 - 1000)/1000 < 0.01$ in average, and the execution time is less than 0.5 [msec].
As the final result, in this operation run, the lexicographic bi-criteria solutions deliver the maximum duration $D_{\text{max}} = 61$ among the entire items (see Eq. (11)), while the single criterion solutions (i.e., the solutions to instances of problem PRIMARY) return the value $D_{\text{max}} = 865$. (Note that generally, $D_{\text{max}} \neq d_{(\ell)\max}^{(1000)}$.) Hence, the empirical results in the figures explain the effectiveness of the lexicographic bi-criteria solutions. We can observe such an effectiveness in another operating run by means of the scatter diagrams.

5. Conclusions

In this paper, we considered a lexicographic bi-criteria combinatorial optimization problem arising in actual food packaging equipments. In each packaging operation, a set of current $n$ items with their weights and priorities, and a target weight $t$ of a package are given. Then, the problem asks to find a subset of the $n$ items so that the total weight of the chosen items is minimized as the primary objective, meeting the target weight constraint, and the total priority of the chosen items is maximized as the second objective. We assumed that all input data are integral. The problem is repeatedly solved during an operating run, in which a large number of packages are produced one by one. The priority of an item is expected to avoid the frequent occurrence of items with longer durations, and it is defined to be the duration of the item. In this paper, utilizing an $O(nts)$ time exact algorithm for the problem, we observed the effectiveness of the lexicographic bi-criteria solutions in an operating run by conducting numerical experiments, which
actually improved the maximum duration in the sampling iterations.

Early combinatorial food packaging equipments have been developed and been progressed by domestic manufactures, while technologies in the packaging equipments have recently attracted much attention of international researches and engineers\(^{(14,15)}\). For future research, mixture packaging and machine setup issues would be still interesting\(^{(11,14)}\).

Acknowledgment

This research was partially supported by JSPS KAKENHI Grant JP16K01241.

References


