Available online at www.sciencedirect.com
ScienceDirect
Computers & Operations Research 36 (2009) 316–324
computers & operations research
www.elsevier.com/locate/cor

L-class enumeration algorithms for a discrete production planning problem with interval resource quantities☆

M.V. Devyaterikova[a], A.A. Kolokolov[b,*], A.P. Kolosov[b]

[a] *Omsk State Technical University, 11, Mir av., Omsk 644050, Russian Federation*
[b] *Omsk Branch of Sobolev Institute of Mathematics of Russian Academy of Sciences, 13, Pevtsov St., Omsk 644099, Russian Federation*

Available online 10 October 2007

Abstract

A discrete production planning problem which may be formulated as the multidimensional knapsack problem is considered, while resource quantities of the problem are supposed to be given as intervals. An approach for solving this problem based on using its relaxation set is suggested. Some L-class enumeration algorithms for the problem are described. Results of computational experiments are presented.

Keywords: Production planning; Integer programming; Interval data; Regular partitions

1. Introduction

The input data of many practical optimization problems are subject to uncertainty. Often it is only known that the data belong to an interval. Therefore the study of the problems with uncertain or interval input data and the design of algorithms for their solving are very important. There are some approaches to solve such problems in discrete optimization. For example, the pessimistic and the optimistic strategies [1], the robust approach [2], the flexible approach [3] deal with the problems where parameters of the objective function are only given as intervals. Some parametric approaches [4] were suggested to solve some integer linear programming (ILP) problems with interval parameters which determine relaxation sets of the problems.

Our approach [5,6] is similar to the one dealing with linear algebraic equations with interval coefficients [7]. Unlike this approach we find a solution of the interval ILP problem. Moreover our techniques permit to solve the ordinary ILP problem approximately.

The proposed approach is applied to the following discrete production planning problem. There are n types of products to be produced and m types of resources needed. Let c_j denote the profit for one unit of the product j, $j = 1, \ldots, n$, and let b_i denote the available quantity of the resource i, $i = 1, \ldots, m$. Values a_{ij} define the consumptions of the resource i for the product j unit, $i = 1, \ldots, m$, $j = 1, \ldots, n$. It is needed to find a production plan $\mathbf{x}^* = (x_1^*, \ldots, x_n^*)$ with integer components which maximizes the total production profit under resource constraints. This problem is also known as

☆ This work is supported by INTAS, project N 03-51-5501.
* Corresponding author.
E-mail address: kolo@iitam.omsk.net.ru (A.A. Kolokolov).

0305-0548/$ - see front matter © 2007 Elsevier Ltd. All rights reserved.
doi:10.1016/j.cor.2007.10.005

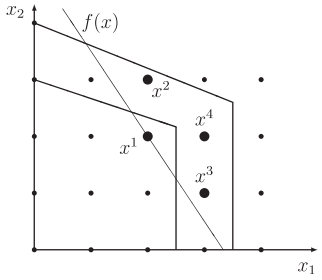

Fig. 1. Solutions to an interval problem.

the multidimensional knapsack problem. The multidimensional knapsack problem with boolean variables is studied as well [8].

The multidimensional knapsack problem also has many other practical applications, such as capital budgeting, cargo load, location in distributed systems, etc. [9–12]. This problem is *NP*-hard [13], there are many exact algorithms and metaheuristics to solve it [8,14,15].

The optimization model of the production planning problem under consideration may be formulated as an ILP problem:

$$\max \quad f(\mathbf{x}) = \sum_{j=1}^{n} \mathbf{c}_j \mathbf{x}_j \tag{1}$$

subject to

$$\sum_{j=1}^{n} a_{ij} x_j \leqslant b_i, \quad i = 1, \ldots, m, \tag{2}$$

$$x_j \geqslant 0, \quad j = 1, \ldots, n, \tag{3}$$

$$x_j \in \mathbf{Z}, \quad j = 1, \ldots, n, \tag{4}$$

where x_j indicates the quantity of the product j, $j = 1, \ldots, n$.

In this paper, the production planning problem with interval values of the resource quantities is investigated. We denote

$$\mathbf{M}(\mathbf{b}) = \left\{ \mathbf{x} \geqslant \mathbf{0} : \sum_{j=1}^{n} a_{ij} x_j \leqslant b_i, i = 1, \ldots, m \right\},$$

where $\mathbf{b} = (b_1, \ldots, b_m)$. Let $b_i^2 \geqslant b_i^1 \geqslant 0$, $i = 1, \ldots, m$. The interval problem is as follows. It is needed to find a point $\mathbf{x}^* \in \mathbf{Z}^n$ satisfying the conditions:

(1) $\mathbf{x}^* \in (\mathbf{M}(\mathbf{b}^2) \cap \mathbf{Z}^n)$,
(2) $f(\mathbf{x}^*) \geqslant f(\mathbf{x})$ for all $\mathbf{x} \in (\mathbf{M}(\mathbf{b}^1) \cap \mathbf{Z}^n)$.

Any point $\mathbf{x}^*$ that satisfies the above conditions is called a solution of the interval problem. This definition is demonstrated in Fig. 1. All points x^1, x^2, x^3 and x^4 are solutions of the interval problem.

It is obvious that any such point $\mathbf{x}^*$ is an optimal solution for the following problem:

$$\max f(\mathbf{x}) = (\mathbf{c}, \mathbf{x}), \quad \mathbf{x} \in (\mathbf{M}(\tilde{\mathbf{b}}) \cap \mathbf{Z}^n),$$

where $\mathbf{c} = (c_1, \ldots, c_n)$, $\tilde{\mathbf{b}}$ is a fixed vector such as $\mathbf{b}^1 \leqslant \tilde{\mathbf{b}} \leqslant \mathbf{b}^2$.

Thus, the interval production planning problem may be solved as the ordinary one with resource product quantities $\mathbf{b}^1$ or $\mathbf{b}^2$. However, such approach does not take into account the specific properties of the problem. It was shown that modifications of well-known algorithms developed specially for the interval problem can reduce computer time

required for solving the problem [16,17]. Besides, the developed algorithms may be used as approximate algorithms for solving problem (1)–(4).

The formulated interval problem is *NP*-hard since it is the ordinary discrete production planning problem in the case of $\mathbf{b}^1 = \mathbf{b}^2$. But in practice most of the interval instances require less computer time than the ordinary ones.

In this paper, a modification of the *L*-class enumeration algorithm for solving the production planning problem with interval input data is presented. It is based on the regular partition of the space $\mathbf{R}^n$ suggested by Kolokolov and called *L*-partition [18]. Regular partitions play an important role in discrete optimization. A lot of results were obtained using these partitions. New classes of cuts were introduced and studied, estimates on the iteration numbers for some cutting plane algorithms and branch-and-bound algorithms were found, structure of some ILP problems was investigated, new promising algorithms were proposed (e.g. for the set covering problem and for the SAT problem), computational experiments were carried out [18]. Recently a new approach to stability analysis of the ILP problems based on the regular partitions has been suggested [19]. The stability of some ILP problems [6] and the stability of some ILP algorithms [20] were established.

The paper is organized as follows. First, the *L*-partition approach is presented. Second, a modification of the basic *L*-class enumeration algorithm for the interval production planning problem is given. Then, applications of the modified algorithm for solving ordinary problem (1)–(4) approximately are discussed. Further, results of computational experiments are presented. Finally, some conclusions and perspectives are considered.

2. The *L*-class enumeration approach

The *L*-partition may be defined as follows. Points $\mathbf{x}, \mathbf{y} \in \mathbf{R}^n$ $(\mathbf{x} \succ \mathbf{y})$ are called *L*-equivalent if there is no $\mathbf{z} \in \mathbf{Z}^n$ such that $\mathbf{x} \succcurlyeq \mathbf{z} \succcurlyeq \mathbf{y}$. Here $\succ, \succcurlyeq$ are the symbols of the lexicographical order. Equivalent points form the classes of the *L*-partition called *L*-classes. Each integer point $\mathbf{z} \in \mathbf{Z}^n$ forms a separate class of the *L*-partition. Other *L*-classes consist of non-integer points and are called fractional *L*-classes. The partition of any set $\mathbf{X}$ into *L*-classes is denoted by $\mathbf{X}/\mathbf{L}$. In the present paper, the following important properties of the *L*-partition are used.

(1) Any fractional *L*-class $\mathbf{V} \in \mathbf{X}/\mathbf{L}$ can be represented as

$$\mathbf{V} = \mathbf{X} \cap \{\mathbf{x} : x_1 = a_1, \ldots, x_{r-1} = a_{r-1}, a_r < x_r < a_r + 1\},$$

where a_j are integer, $j = 1, \ldots, r$, $1 \leqslant r \leqslant n$. The number r is called the rank of the *L*-class and denoted by $\mathrm{rank}(\mathbf{V})$. $\mathrm{Rank}(\mathbf{z})$ is assumed to be equal $n + 1$ for any integer point $\mathbf{z}$.

(2) Let $\mathbf{X}, \mathbf{X}'$ be non-empty sets from $\mathbf{R}^n$. The set $\mathbf{X}$ is called lexicographically greater than $\mathbf{X}'(\mathbf{X} \succ \mathbf{X}')$ if $\mathbf{x} \succ \mathbf{x}'$ for any $\mathbf{x} \in \mathbf{X}$ and $\mathbf{x}' \in \mathbf{X}'$. This relation is a linear order on $\mathbf{X}/\mathbf{L}$ for any set $\mathbf{X} \subseteq \mathbf{R}^n$. If $\mathbf{X}$ is a bounded set, then $\mathbf{X}/\mathbf{L}$ can be represented as follows:

$$\mathbf{X}/\mathbf{L} = \{\mathbf{V}_1, \ldots, \mathbf{V}_p\}, \quad \mathbf{V}_i \succ \mathbf{V}_{i+1}, \quad i = 1, \ldots, p-1.$$

There are $(n + 1)$ types of *L*-classes in R^n. For example, there are three types of them in R^2: integer points, vertical intervals and vertical stripes. In Fig. 2 *L*-classes V_1 and V_5 are vertical stripes with rank 1, V_2 and V_4 are vertical intervals with rank 2, and V_3 is integer point of rank 3.

Let us consider the idea of the *L*-class enumeration method for problem (1)–(4). The main step of the basic algorithm is to pass from the current *L*-class of the relaxation set $\mathbf{M}(\mathbf{b})$ to another one according to the lexicographically decreasing order. The record value of the objective function $f(\mathbf{x})$ is also taken into account.

The algorithm generates a sequence $\mathbf{S}$ of points $\mathbf{x}^{(t)} \in \mathbf{M}(\mathbf{b})$ with the following properties:

(1) $\mathbf{x}^{(t)} \succ \mathbf{x}^{(t+1)}$, $t = 1, 2, \ldots$;
(2) all $\mathbf{x}^{(t)}$ belong to different *L*-classes;
(3) if $\mathbf{M}(\mathbf{b}) \cap \mathbf{Z}^n \neq \emptyset$, then $\mathbf{S}$ contains a subsequence of integer points $\mathbf{Q} = \{\mathbf{z}^{(t_k)}, k = 1, \ldots, q\}$ such that $f(\mathbf{z}^{(t_{k+1})}) > f(\mathbf{z}^{(t_k)})$, $k = 1, \ldots, q-1$ as $q > 1$.

A version of the *L*-class enumeration algorithm where the enumeration process is started with the lexicographically maximal point $\mathbf{x}^{(1)} \in \mathbf{M}(\mathbf{b})$ is considered. We denote it by *LE*. The current points $\mathbf{x}^{(t)}$ are constructed by finding

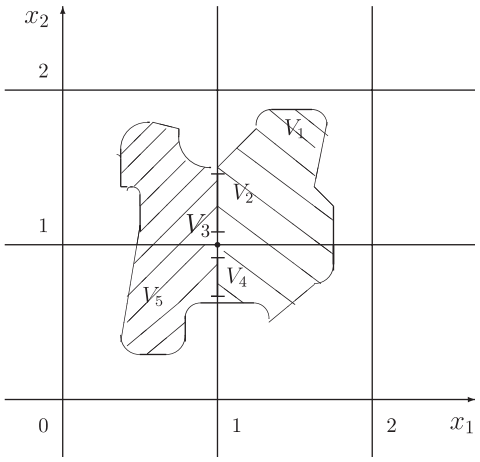

Fig. 2. L-classes in R^2.

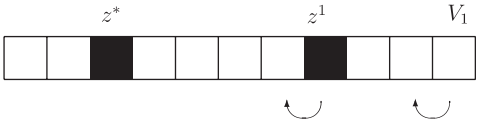

Fig. 3. Enumeration on the tape of L-classes.

the lexicographical maximums of auxiliary linear programming subproblems. The search for these maxima may be implemented, for example, by the lexicographical dual simplex method. The algorithm completes enumeration if it cannot find a new L-class. After a finite number of steps the algorithm finds an optimal solution of problem (1)–(4).

In the view of the linear order on the set of L-classes the L-class enumeration may be illustrated with the following "tape" (see Fig. 3). Each box of the tape corresponds to a separate L-class, the white boxes represent fractional L-classes and the black boxes represent integer points.

Note that Algorithm *LE* also may be applied to the boolean multidimensional knapsack problem and some other ILP problems.

3. A modified L-class enumeration algorithm for the problem with interval data

In this section a modification of the basic L-class enumeration algorithm for solving the interval production planning problem is described. It is denoted by *LEM*. The distinction between the basic algorithm and Algorithm *LEM* is the following. Algorithm *LEM* starts with the set $\mathbf{M}(\mathbf{b}^1)$. If a current linear programming subproblem is infeasible, Algorithm *LEM* increases the right sides of the constraints of the subproblem (not exceeding $\mathbf{b}^2$) to obtain its optimal solution. In other words, the algorithm passes to the tape of L-classes of the relaxation set $\mathbf{M}(\tilde{\mathbf{b}})$, $\mathbf{b}^1 \leqslant \tilde{\mathbf{b}} \leqslant \mathbf{b}^2$. The meaning of this procedure is to find a new integer point faster, update the record value of the objective function and reduce the CPU time required. After this procedure the algorithm returns to the L-class enumeration of the relaxation set $\mathbf{M}(\mathbf{b}^1)$. In all other respects the basic L-class enumeration algorithm and Algorithm *LEM* are the same. The idea of the modified algorithm is illustrated with Fig. 4.

Before describing Algorithm *LEM* let us define

$$\hat{\delta} = \inf\{|f(\mathbf{z}') - \mathbf{f}(\mathbf{z}'')|\colon \mathbf{z}', \mathbf{z}'' \in (\mathbf{M}(\mathbf{b}^2) \cap \mathbf{Z}^n) \text{ and } f(\mathbf{z}') \neq \mathbf{f}(\mathbf{z}'')\}$$

when $\mathbf{M}(\mathbf{b}^2)$ contains at least two integer points. In this case $\hat{\delta} > 0$ since $\mathbf{M}(\mathbf{b}^2)$ is a closed bounded set. Otherwise $\hat{\delta}$ may be arbitrary. For example, if all coefficients of the objective function are integer-valued, then $\hat{\delta}$ may be taken as 1. We choose $0 < \delta \leqslant \hat{\delta}$.

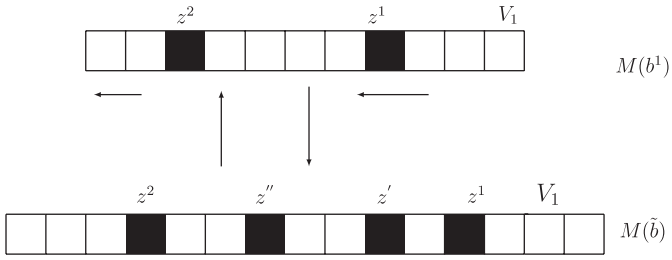

Fig. 4. The modification of *L*-class enumeration.

Take $rec = -\infty$ as the initial record value of the objective function.

Algorithm *LEM*

Step 1. Solve the problem

$$\max f(\mathbf{x}) = (\mathbf{c}, \mathbf{x}), \mathbf{x} \in \mathbf{M}(\mathbf{b}^1).$$

If the optimal solution of the problem is integer,
then the initial interval problem is solved. Otherwise go to step 2.

Step 2. Solve the problem

$$\max f(\mathbf{x}) = (\mathbf{c}, \mathbf{x}), \mathbf{x} \in \mathbf{M}(\mathbf{b}^2).$$

If the optimal solution of the problem is integer,
then the initial interval problem is solved. Otherwise go to step 3.

Step 3. Find $\mathbf{x}' = lexmax\ \mathbf{M}(\mathbf{b}^1)$. Two cases are possible.
3.1. If $\mathbf{x}' \in \mathbf{Z}^n$, set $p = n + 1$, $\mathbf{x}'' = \mathbf{x}'$,
calculate the new record $rec = f(\mathbf{x}')$ and go to step 5.
3.2. If $\mathbf{x}' \notin \mathbf{Z}^n$, go to step 4.

Step 4. Search for the next *L*-class ("move down").
Set $\mathbf{x}'' = \mathbf{x}'$. Find

$$p = \min\{j : x''_j \neq \lfloor x''_j \rfloor, j = 1, \ldots, n\}.$$

Solve the subproblem:

$$\text{find } \mathbf{x}' = lexmax\{\mathbf{x} \in \mathbf{M}(\mathbf{b}^1) : f(\mathbf{x}) \geqslant \mathbf{rec} + \delta,$$
$$x_1 = x''_1, \ldots, x_{p-1} = x''_{p-1}, x_p \leqslant \lfloor x''_p \rfloor\}.$$

The following cases are possible.
4.1. If the subproblem is not solvable, go to step 6.
4.2. If the subproblem is solvable and $\mathbf{x}' \in \mathbf{Z}^n$,
set $p = n + 1$, $\mathbf{x}'' = \mathbf{x}'$, update record $rec = f(\mathbf{x}')$, go to step 5.
4.3. If the subproblem is solvable and $\mathbf{x}' \notin \mathbf{Z}^n$, go to step 4.

Step 5. Search for the next *L*-class ("move up").
Set $\varphi = \max\{j : 0 < j < p, x''_j > 0\}$.
If there is no such φ, go to step 8.
Otherwise solve the subproblem:

$$\text{find } \mathbf{x}' = lexmax\{\mathbf{x} \in \mathbf{M}(\mathbf{b}^1) : f(\mathbf{x}) \geqslant \mathbf{rec} + \delta,$$
$$x_1 = x''_1, \ldots, x_{\varphi-1} = x''_{\varphi-1}, x_\varphi \leqslant x''_\varphi - 1\}.$$

The following cases are possible.

5.1. If the subproblem is not solvable, go to step 7.

5.2. If the subproblem is solvable and $\mathbf{x}' \in \mathbf{Z}^n$, set $p = n + 1$, $\mathbf{x}'' = \mathbf{x}'$, update the record $rec = f(x')$, go to step 5.

5.3. If the subproblem is solvable and $\mathbf{x}' \notin \mathbf{Z}^n$, go to step 4.

Step 6. Search for the next L-class ("move down").

Find a vector $\tilde{\mathbf{b}}$, $\mathbf{b}^1 \leqslant \tilde{\mathbf{b}} \leqslant \mathbf{b}^2$, such that the following subproblem is solvable:

$$\text{find } \mathbf{x}' = lexmax\{\mathbf{x} \in \mathbf{M}(\tilde{\mathbf{b}}) : f(\mathbf{x}) \geqslant \mathbf{rec} + \delta,$$
$$x_1 = x_1'', \ldots, x_{p-1} = x_{p-1}'', x_p \leqslant \lfloor x_p'' \rfloor\}.$$

The following cases are possible.

6.1. If such $\tilde{\mathbf{b}}$ does not exist and $p = 1$, go to step 8.

6.2. If such $\tilde{\mathbf{b}}$ does not exist and $p > 1$, go to step 5.

6.3. If the subproblem is solvable and $\mathbf{x}' \in \mathbf{Z}^n$, set $p = n + 1$, $\mathbf{x}'' = \mathbf{x}'$, update record $rec = f(\mathbf{x}')$, go to step 5.

6.4. If the subproblem is solvable and $\mathbf{x}' \notin \mathbf{Z}^n$, go to step 4.

Step 7. Search for the next L-class ("move up").

Find a vector $\tilde{\mathbf{b}}$, $\mathbf{b}^1 \leqslant \tilde{\mathbf{b}} \leqslant \mathbf{b}^2$, such that the following subproblem is solvable:

$$\text{find } \mathbf{x}' = lexmax\{\mathbf{x} \in \mathbf{M}(\tilde{b}) : f(\mathbf{x}) \geqslant \mathbf{rec} + \delta,$$
$$x_1 = x_1'', \ldots, x_{\varphi-1} = x_{\varphi-1}'', x_\varphi \leqslant x_\varphi'' - 1\}.$$

The following cases are possible.

7.1. If such $\tilde{\mathbf{b}}$ does not exist and $\varphi = 1$, go to step 8.

7.2. If such $\tilde{\mathbf{b}}$ does not exist and $\varphi > 1$, go to step 5.

7.3. If the subproblem is solvable and $\mathbf{x}' \in \mathbf{Z}^n$, set $p = n + 1$, $\mathbf{x}'' = \mathbf{x}'$, update the record $rec = f(\mathbf{x}')$, go to step 5.

7.4. If the subproblem is solvable and $\mathbf{x}' \notin \mathbf{Z}^n$, go to step 4.

Step 8. The solution process is terminated.

The best found integer point is a solution for the interval problem.

Steps 1–3 are preliminary and they are performed only once. The main iterations of the algorithm are fulfilled in steps 4–7. The kind of linear problems in these steps is determined by the representation of an L-class (see property 1). In steps 4 and 5 the lexicographical maximal point of the next L-class (according to the lexicographical decreasing order) of the relaxation set $\mathbf{M}(\mathbf{b}^1)$ is being found, but in step 4 the mentioned L-class has a rank greater or equal to the rank of the current one ("move down"), and in step 5 it has a rank less than the rank of the current one ("move up"). The same procedure is fulfilled in steps 6 and 7 but with relaxation set $\mathbf{M}(\tilde{\mathbf{b}})$. The jump to step 8 happens when L-class enumeration is terminated.

Note that the basic L-class enumeration algorithm may be considered as an approximate algorithm in the case $\delta > \hat{\delta}$.

It is easy to show that Algorithm *LEM* finds a solution of the considered interval problem after a finite number of steps. Besides, the algorithm after some modifications may be applied to a wider class of interval optimization problems.

4. Approximate algorithms

Using the described techniques for interval production planning problem we can develop some approximate algorithms for ordinary problem (1)–(4). Two types of algorithms based on Algorithm *LEM* were implemented for this

problem. Denote them by $LEM1$ and $LEM2$, respectively.

The first algorithm uses a contraction of the relaxation set $\mathbf{M}(\mathbf{b})$. It takes $\mathbf{b}$ as $\mathbf{b}^2$ and $(1-\varepsilon)\mathbf{b}$ as $\mathbf{b}^1$ where $\varepsilon \in (0, 1)$. Some L-classes and integer points from $\mathbf{M}(\mathbf{b})$ may be missed during the solving process. Therefore, the value ε must be small enough to obtain a good solution.

The second algorithm is based on an extension of the relaxation set $\mathbf{M}(\mathbf{b})$. In this case $\mathbf{b}^1 = \mathbf{b}$, $\mathbf{b}^2 = (1+\varepsilon)\mathbf{b}$, $\varepsilon \in (0, 1)$. The algorithm in general may obtain an infeasible integer point as a result. Therefore, the best found feasible integer point is taken as an approximate solution of the considered problem. The value ε also must be small to obtain a good approximate solution.

5. Computational experiments

Now we present the results of computational testing of the suggested algorithms. The behavior of Algorithm *LEM* on the interval production planning problems was investigated. The possibility of application of Algorithm *LEM* to finding an approximate solution of problem (1)–(4) was studied as well.

It is assumed that $\mathbf{b}^1 = (1-\varepsilon)\mathbf{b}$ and $\mathbf{b}^2 = \mathbf{b}$ where $\varepsilon \in (0, 1)$ while solving the interval problem. Hence a solution of the interval problem may be taken as an approximate solution of problem (1)–(4) and vice versa. In this case the Algorithms *LEM* and $LEM1$ are the same.

Algorithm *LEM* was tested on a number of series of problem instances with randomly generated integer input data drawn for the uniform distribution. Each series consisted of 30 instances. The number of variables n and the number of constraints m ranged from 40 to 200 and from 10 to 300, respectively. The values of profits c_j, $j = 1, \ldots, n$ were taken from the intervals [1, 10] and [1, 100]. The instances with $c_j = 1$, $j = 1, \ldots, n$ were considered as well. The values of a_{ij}, $i = 1, \ldots, m$, $j = 1, \ldots, n$ were taken from the interval [0, 10] and from the set {0, 1}. The available quantities of resources b_i were constructed by the formula $b_i = [(\sum_{j=1}^{n} a_{ij})/2]$, $i = 1, \ldots, m$ and in other ways. The experiments were carried out with different values of ε(0.005, 0.01, 0.02, 0.05). The tests were performed on an Pentium-4 CPU 2.8 GHz.

In Tables 1 and 2 the average CPU times required to solve the instances and the average and maximal relative errors of the optimal values of the objective function are indicated. Here t_{opt} is the average CPU time of solving problem (1)–(4) by Algorithm *LE* for each series, t_ε is the average CPU time of solving the interval problem by Algorithm *LEM* for each series with the given ε, $\Delta_\varepsilon^{\text{av}}$ and $\Delta_\varepsilon^{\max}$ are the average and maximal relative errors of the optimal values of the objective function for each series with the given ε. The CPU time is given in seconds.

The experiments show that if n is considerably greater than m, the instances are solved very fast by Algorithm *LEM*. For example, if $\varepsilon = 0.01$, the averaged t_{opt} is 166 times greater than t_ε and the averaged t_{opt} is 1886 times greater than t_ε in the case $\varepsilon = 0.02$. For most instances if ε is greater, then required computer time is smaller. When Algorithm *LEM* is

Table 1
Results for randomly generated series in the case $n > m$

$m \times n$	t_{opt}	$t_{0.01}$	$\Delta_{0.01}^{\text{av}}$	$\Delta_{0.01}^{\max}$	$t_{0.02}$	$\Delta_{0.02}^{\text{av}}$	$\Delta_{0.02}^{\max}$
20 × 100	39.8	3.1	0.008	0.010	0.3	0.015	0.020
10 × 150	14.5	0.2	0.007	0.009	0.0	0.013	0.019
20 × 150	119.3	1.4	0.007	0.011	0.1	0.015	0.020
10 × 200	21.1	0.1	0.007	0.010	0.0	0.009	0.017
20 × 200	392.2	1.1	0.008	0.010	0.1	0.014	0.019

Table 2
Results for randomly generated series in the case $n < m$

$m \times n$	t_{opt}	$t_{0.01}$	$\Delta_{0.01}^{av}$	$\Delta_{0.01}^{max}$	$t_{0.02}$	$\Delta_{0.02}^{av}$	$\Delta_{0.02}^{max}$
100×50	30.3	9.4	0.009	0.017	4.1	0.014	0.022
200×50	68.2	27.7	0.009	0.018	10.4	0.014	0.023
300×40	21.7	15.8	0.009	0.022	10.1	0.016	0.029
300×50	114.7	41.7	0.010	0.019	19.8	0.014	0.030

Table 3
Results for OR library instances

Series	$m \times n$	t_ε	ε	Δ_ε^{av}	Δ_ε^{max}
mknapcb1	5×100	3.3	0.010	0.003	0.006
mknapcb2	5×250	7.0	0.005	0.004	0.016
mknapcb3	5×500	6.6	0.005	0.003	0.004
mknapcb4	10×100	28.0	0.010	0.006	0.010
mknapcb5	10×250	24.7	0.010	0.004	0.006
mknapcb6	10×500	8.3	0.010	0.004	0.006
mknapcb7	30×100	690.0	0.010	0.006	0.010

considered as the approximate algorithm the relative error of the total profit for the mentioned instances is smaller than ε (it is not true in some computer experiments). Moreover, Algorithm *LEM* works 30 times faster on average than the basic *L*-class enumeration algorithm finding the same approximate solution in the case $\varepsilon = 0.01$ and 100 times faster in the case $\varepsilon = 0.02$.

On the contrary, the results are not so good in the case $n < m$. But the interval problems are solved faster than the ordinary problems even in this case.

The comparison of Algorithms $LEM1$ and $LEM2$ showed that the $LEM1$ was better for all instances.

Some instances from the OR library were also used (http://people.brunel.ac.uk/mastjjb/jeb/info.html). They are difficult multidimensional boolean knapsack problems. For many of these problems the best solutions are only known. Computational experiments were performed on the series mknapcb1-mknapcb7. Algorithm $LEM1$ was applied to obtain approximate solutions for these instances. In most cases Algorithm $LEM1$ found the mentioned best solutions, in other cases it found approximate solutions very close to the best ones. The behavior of Algorithm $LEM1$ was in general the same on the randomly generated instances and on the OR library ones. The results of the computational experiment for the OR library instances are presented in Table 3.

6. Conclusion

The proposed approach seems to be perspective enough for solving discrete production planning problem with interval input data. Besides, the promising algorithms for finding approximate solutions of the ordinary production planning problem were developed. The described algorithms can be more efficient if valid cuts and some heuristics are used.

In this paper, the *L*-class enumeration method for the interval discrete production planning problem is considered in details. But the developed scheme may be applied for other algorithms which use relaxation sets of the studied problem (e.g. branch-and-bound algorithm). The suggested approach also may be adapted to other integer programming problems with interval input data.

References

[1] Libura M. Integer programming problems with inexact objective function. Control and Cybernetics 1980;9(4):189–202.
[2] Kouvelis P, Yu G. Robust discrete optimization and its applications. Netherlands: Kluwer Academic Publishers; 1997.
[3] Manjoub A, Mihelic J, Rapine C, Robic B. k-center problem with uncertainty: flexible approach. In: Proceedings of discrete optimization methods in production and logistics (DOM-2004). Omsk: Nasledie Dialog-Sibir; 2004. p. 75–80.

[4] Sergienko IV, Kozeratskaya LN, Lebedeva TT. Study of stability and parametrical analysis of discrete optimization problems. Ukraine: Naukova dumka; 1995 [in Russian].
[5] Devyaterikova MV, Kolokolov AA. *L*-class enumeration algorithms for knapsack problem with interval data. In: Book of abstracts. International conference on operations research OR2001. Duisburg, 2001. p. 118.
[6] Devyaterikova MV, Kolokolov AA. Analysis of *L*-structure stability of convex integer programming problems. In: Operations research proceedings. Berlin: Springer; 2000. p. 49–54.
[7] Neumaier A. Interval methods for systems of equations. UK: Cambridge University Press; 1990.
[8] Martello S, Toth P. Knapsack problem: algorithms and computer implementations. USA: Wiley; 1990.
[9] Akcay Y, Xu SH. Joint inventory replenishment and component allocation optimization in an assemble-to-order system. Management Science 2004;50:99–116.
[10] Lin EY-H. A bibliographical survey on some well-known non-standard knapsack problems. INFOR 1998;36(4):274–317.
[11] Lu LL, Chiu SY, Cox LA. Optimal project selection: stochastic knapsack with finite time horizon. Operations Research 1999;50:645–50.
[12] de Vries S, Vokra RV. Combinatorial auctions: a survey. Discussion Paper 1296, The Center for Mathematical Studies in Economics and Management Science, Northwestern University; 2001.
[13] Garey MR, Johnson DS. Computers and intractability: a guide to the theory of NP-completeness. San Francisco: W.H. Freeman; 1979.
[14] Caprara A, Kellerer H, Pferschy U, Pisinger D. Approximation algorithms for knapsack problems with cardinality constraints. European Journal of Operational Research 2000;123:333–45.
[15] Chu PC, Beasley JE. A genetic algorithm for the multidimensional knapsack problem. Journal of Heuristics 1998;4:63–86.
[16] Devyaterikova MV, Kolokolov AA. *L*-class enumeration algorithm for one interval production planning problem. In: A proceedings volume from the 12th IFAC symposium. Oxford: Elsevier; 2006. p. 9–13.
[17] Devyaterikova MV, Kolokolov AA. *L*-class enumeration algorithms for the knapsack problem with interval input data. Omsk State University, Russia, Preprint; 2001 [in Russian].
[18] Kolokolov AA. Regular partitions and cuts in integer programming. In: Discrete analysis and operations research. Netherlands: Kluwer Academic Publishers; 1996. p. 59–79.
[19] Kolokolov AA, Devyaterikova MV. Analysis of the stability of the *L*-partition of sets in Euclidean space. Discrete Analysis and Operations Research 2000;2(7):47–53 [in Russian].
[20] Kolokolov AA, Devyaterikova MV. Stability analysis of some discrete optimization algorithms. In: Proceedings of discrete optimization methods in production and logistics (DOM-2004). Omsk: Nasledie Dialog-Sibir; 2004. p. 180–4.