# Integer Programming Formulations For The Frobenius Problem

Imdat Kara*, Halil Ibrahim Karakas
Department of Industrial Engineering, Department of Finance and Banking
Baskent University
Baglica Campus, 06790, Etimesgut, Ankara, Turkey
Turkey
ikara@baskent.edu.tr, karakas@baskent.edu.tr
http://www.baskent.edu.tr/~ikara/, https://www.baskent.edu.tr/~karakas/

***Abstract:* - The Frobenius number of a set of relatively prime positive integers $a_1, a_2, \ldots, a_n$ such that $a_1 < a_2 < \ldots < a_n$, is the largest integer that can not be written as a nonnegative integer linear combination of the given set. Finding the Frobenius number is known as the *Frobenius problem*, which is also named as the *coin exchange problem* or the *postage stamp problem*. This problem is closely related with the equality constrained *integer knapsack problem*. It is known that this problem is NP-hard. Extensive research has been conducted for finding the Frobenius number of a given set of positive integers. An exact formula exists for the case $n = 2$ and various formulas have been derived for all special cases of n = 3. Many algorithms have been proposed for $n \geq 4$.**

**As far as we are aware, there does not exist any integer programming approach for this problem which is the main motivation of this paper. We present four integer linear programming formulations about the Frobenius number of a given set of positive integers. Our first formulation is used to check if a given positive integer is the Frobenius number of a given set of positive integers. The second formulation aims at finding the Frobenius number directly. The third formulation involves the residue classes with respect to the least member of the given set of positive integers, where a residue table is computed comprising all values modulo that least member, and the Frobenius number is obtained from there. Based on the same approach underlying the third formulation, we propose our fourth formulation which produces the Frobenius number directly.**

**We demonstrate how to use our formulations with several examples. For illustrative purposes, some computa-tional analysis is also presented.**

***Key-Words:* - Frobenius problem; Frobenius numbers; integer programming; modelling.**

## I. INTRODUCTION

Let $a_1, a_2, \ldots, a_n$ be fixed positive integers (PIs) such that $a_1 < a_2 < \cdots < a_n$ and the greatest common divisor of them is equal to 1. A non-negative integer linear combination (NILC) of $a_1, a_2, \ldots, a_n$ is written as,

$$N = \sum_{i=1}^{n} x_i\, a_i \qquad (1)$$

where $x_1, x_2, \ldots, x_n$ are nonnegative integers. It is well known that there are only finitely many positive integers which can not be expressed as a NILC of $a_1, a_2, \ldots, a_n$.

Ferdinand Georg Frobenius (1849-1947) used to ask to the students in his lecture to find the largest integer that can not be expressed as a NILC of $a_1, a_2, \ldots, a_n$ (Alfonsin [2]). For this reason, the largest integer that cannot be expressed as a NILC of $a_1, a_2, \ldots, a_n$ is named as the Frobenius number (FN) of $a_1, a_2, \ldots, a_n$ and it is denoted by $F(a_1, a_2, \ldots, a_n)$.
Thus, all the integers greater than the Frobenius number $F = F(a_1, a_2, \ldots, a_n)$ can be expressed as a NILC of $a_1, a_2, \ldots, a_n$. That is to say the equation

$$\sum_{i=1}^{n} x_i a_i = F + j \qquad (2)$$

has nonnegative integer solutions for each $j \geq 1$.

Finding the FN of a given set of PIs is named as the *Frobenius problem* which is also known as the *coin exchange problem* or the *postage stamp problem* (Tripathi [26]). This problem is also related to the equality constrained *integer knapsack problem* (Böcker and Liptak [7]). Sylvester [23] showed that

$$F(a_1, a_2) = a_1 a_2 - a_1 - a_2 \qquad (3)$$

Since 1884, extensive researches have been conducted for finding exact formulas for the FN of a set of three or more integers. Brauer [8] found the Frobenius number for consecutive integers. Roberts [20] developed a formula for arithmetic sequences. There are few cases for which the Frobenius number has been exactly determined.

Curtis [10] showed that no closed formula can be found for all $n > 2$. Formulas covering all cases for three integers have been given recently by Tripathi [26]. Alfonsin [3] proved that the Frobenius problem is NP-hard for $n > 2$.

Research on the Frobenius problem has mainly been on finding bounds for the Frobenius number and algorithmic aspects. In 1975, Lewin [15] proposed an algorithm for finding the FN of a given set of PIs. Later, Owens [19] showed that Lewin's algorithm produces wrong results. Since then, considerable amount of algorithms and approaches have been developed for various cases (Böcker and Liptak [7]; Trimm [25]; Einstein et al [12]). Beihoffer et all [6] developped graph theory based algorithms and conducted detailed computational analysis in 2005. Later, Roune [21] developed an algorithm that produces the Frobenius numbers of some sets of thousand-digit PIs. Ong and Ponomarenko [18] developed some formulas for the FN of geometric sequences.

Applications of the Frobenius problem occur in several areas like number theory, automata theory, sorting algorithms, petry nets, etc. Alfonsin [2] surveys all the related references, outlines most of the applications and indicates open questions about the Frobenius problem.

The Frobenius problem is still attracting researchers. Recently, two Ph.D.thesis appeared on this subject (Mohammed [16]; Navarro [17]). There is a web site [24] named as "The On-Line Encyclopedia of Integer Sequences" whose address is http://oeis.org.
In this site, one can find the Frobenius numbers of some special sets of positive integers such as "Frobenius numbers of arithmetic sequences up to 8 successive numbers", "Frobenius numbers of three successive primes" etc. There is a GAP package (Delgado et al [11]) which can be used to compute the FN of any set of PIs. The program is extremely fast.

From the mathematical programming point of view, the Frobenius problem is closely related with integer programming. Vizvari [27] produces upper bounds and gives exact solutions for some subproblems by applying integer programming. Krawczyk and Paz [14] propose close bounds by solving a series of integer linear programs. Aardal and Lenstra [1] use Frobenius numbers to create infeasible instances for integer knapsack problems. Aliev and Henk [4, 5] discover the average behaviour of the Frobenius number based on integer knapsacks. There is also a close relationship between integer knapsack feasibility and Frobenius numbers (Hansen and Ryan [13]; Ryan [22]).

The remarkable improvement in hardware and software technology and widespread availability of commercial optimisation software will allow us to solve many models easily in the near future. To the best of our knowledge, there is no approach and/or algorithm for finding the FN directly by using integer programming formulations which is the main motivation of this study. Our aim has been to develop some user friendly mathematical models for finding the FN of a given set of positive integers that can be executed by using any optimizer. Hopefully, we developed very useful mathematical models. Our contributions may be summarized as follows:

1. We propose integer linear programming formu-lations for checking the necessary and sufficient conditions for a positive integer to be the FN of a given set of PIs.
2. We present a general integer linear prog-ramming model for finding the FN of a given set of relatively prime positive integers.
3. In order to reduce the solution time of the general model, we rearranged the model with rgard to the smallest integer in the given set. So then, we are able to find the FN very easily.

In the following section, we propose two formulations, discuss their usage and give some examples. In section 3, we generalize our approach by proposing two more formulations so that one can find the FN of a given set of three or more PIs very
easily by using an optimizer. In the last section, we summarize our findings and express some remarks for further researches.

## II. TEST FOR THE FROBENIUS NUMBER

In what follows, $a_1, a_2, \ldots, a_n$ denote PIs such that $a_1 < a_2 < \cdots < a_n$ and the greatest common divisor of them is equal to 1.

Given a positive integer $\alpha$, the following integer linear programming model can be used to check if the equation

$$x_1a_1 + x_2a_2 + \cdots + x_na_n = \alpha \qquad (4)$$

has a solution in the set of nonnegative integers.

$M_1$:
$\min p$
$s.t.$
$a_1x_1 + a_2x_2 + \cdots + a_nx_n - p = \alpha$
$x_i \geq 0$ and integer; $i = 1, \ldots, n; p \geq 0$ and integer.

Since the equation (4) has a solution when $\alpha$ is large enough, the model $M_1$ has an optimal solution. Thus we can state the following proposition.

**Proposition 2.1.** Let the optimal solution of $M_1$ be denoted as $\bar{x}_1, \bar{x}_2, \ldots, \bar{x}_n, \bar{p}$ . Then the equation (4) has a solution in the set of nonnegative integers if and only if $\bar{p} = 0$.

All the examples given in this paper are solved with CPLEX 12.8 by using an Intel Core i7-3630QM CPU 2.40 GHz and 16 GB RAM computer.

**Example 2.2.** Consider the integers 11, 882, 1017, 1218 and let $\alpha$ = 3359. The corresponding model $M_1$ is

$$\begin{aligned}&\min p\\ &s.t.\\ &11x_1 + 882x_2 + 1017x_3 + 1218x_4 - p = 3359\\ &x_i \geq 0 \text{ and integer; } i = 1, 2, 3, 4; p \geq 0 \text{ and}\\ &\text{integer}\end{aligned}$$

This model gives the optimal solution

$$\bar{x}_1 = 145, \bar{x}_2 = 2, \bar{x}_3 = \bar{x}_4 = 0, \bar{p} = 0 .$$

Hence 3359 can be expressed as a NILC of 11, 882, 1017, 1218.

**Example 2.3 (Owens [19]).** Consider the integers 34, 37, 38, 40, 43 and let $\alpha$ = 163. The correspond-ing model $M_1$ is:

$$\begin{aligned}&\min p\\ &s.t.\\ &34x_1 + 37x_2 + 38x_3 + 40x_4 + 43x_5 - p = 163\\ &x_i \geq 0 \text{ and integer; } i = 1, 2, 3, 4, 5; p \geq 0 \text{ and}\\ &\text{integer}\end{aligned}$$

The value of $p$ in the optimal solution of the above model is $\bar{p} = 0$. Therefore, 163 can be expressed as a NILC of 34, 37, 38, 40, 43 and thus it is not the FN of 34, 37, 38, 40, 43. As mentioned before, Lewin[15]'s algorithm produces incorrectly 163 as the Frobenius number of 34, 37, 38, 40, 43 (Owens [19]).

An immediate corollary to Proposition 2.1 is the following.

**Corollary 2.4.** If $\alpha$ is the Frobenius number of $a_1, a_2, \dots, a_n$, then the value of $p$ in the optimal solution of M1 is $\bar{p} \geq 1$ .

The above corollary gives a necessary condition for $\alpha$ to be the FN of a given set of PIs. Obviously, this condition is not sufficient for $p$ to be the FN. For instance, if one takes the right hand side constant of the model given in Example 2.3 above as equal to 165, the value of $p$ in the optimal solution will be equal to 1. Thus 165 can not be written as a NILC of 34, 37, 38, 40, 43, but as Owens [19] shows and as we will show in Example 3.1, the Frobenius number of the set {34, 37, 38, 40, 43} is 175 not 165. So, we need more tests for finding the FN.

To get a sufficient condition for an integer to be the FN of $a_1, a_2, \dots, a_n$, we use the following theorem.

**Theorem 2.5.** Let $\alpha$ be a positive integer such that $\alpha + j$ can be expressed as a NILC of $a_1, a_2, \dots, a_n$ for each $j = 1, 2, \dots, a_1$. Then the Frobenius num-ber of $a_1, a_2, \dots, a_n$ is not larger than $\alpha$.

**Proof of Theorem 2.5.** It suffices to prove that every integer larger than $\alpha$ can be expressed as a NILC of $a_1, a_2, \dots, a_n$. In fact, let $k$ be an integer such that $k \geq \alpha + 1$. Divide $k - (\alpha + 1)$ by $a_1$:

$$k - (\alpha + 1) = qa_1 + r, q \geq 0, 0 \leq r \leq a_1 - 1$$

Then

$$k = qa_1 + (\alpha + 1 + r)$$

where $1 \leq 1 + r \leq a_1$. This shows that $k$ can be expressed as a NILC of $a_1, a_2, \dots, a_n$.

The next corollary leads to develop another integer linear programming model ($M_2$ below) which gives the Frobenius number directly.

**Corollary 2.6** If $\alpha$ is the smallest integer satisfying the conditions in Theorem 2.5, then $\alpha$ is the Frobenius number of $a_1, a_2, \dots, a_n$.

Hence the value of $\alpha$ in the optimal solution of the following model $M_2$ gives the Frobenius number of $a_1, a_2, \dots, a_n$.

$$\begin{aligned}&M_2:\\ &\min \alpha\\ &\textstyle\sum_{i=1}^{n} a_i x_{ij} - \alpha = j\\ &x_{ij} \geq 0 \text{ and integer; } i = 1, \dots, n; j = 1, \dots, a_1\\ &\alpha \geq 0 \text{ and integer}\end{aligned}$$

Note that the number of decision variables in the model $M_2$ is $na_1 + 1$ while the number of constraints is $a_1$.

**Example 2.7** We apply $M_2$ to the set {4, 63, 73, 111}. The explicit form of the corresponding $M_2$ model is written as

$$\begin{aligned}&\min \alpha\\ &s.t.\\ &4x_{11} + 63x_{21} + 73x_{31} + 111x_{41} - \alpha = 1\\ &4x_{12} + 63x_{22} + 73x_{32} + 111x_{42} - \alpha = 2\\ &4x_{13} + 63x_{23} + 73x_{33} + 111x_{43} - \alpha = 3\\ &4x_{14} + 63x_{24} + 73x_{34} + 111x_{44} - \alpha = 4\\ &x_{ij} \geq 0 \text{ and integer; } i, j = 1, 2, 3, 4\\ &\alpha \geq 0 \text{ and integer}\end{aligned}$$

The optimal value of $\alpha$ turns out to be 122. Thus, the Frobenius number of the given set of PIs is 122. When we delete 111 from the set {4, 63, 73, 111} and solve the model for the set {4, 63, 73}, we get the same number 122 as the FN of the set {4, 63, 73}. So F(4, 63, 73, 111) = F(4, 63, 73) = 122. This is because 111 can be expressed as a NILC of 4, 63, 73: 111 = 12 · 4 + 63.

As the above example shows, if any of the integers $a_1, a_2, \dots, a_n$, say $a_k$, can be expressed as a NILC of the remaining ones, then $\{a_1, a_2, \dots, a_n\}$ and the set obtained by deleting $a_k$ from it have the same FN. We have used the model $M_2$ to find the FN of various choices of $a_1, a_2, \dots, a_n$ where $a_1$ is not very large. Since dimension of the model depends directly upon the value of $a_1$, it may take a lot of time to find the FN when $a_1$ gets larger. The state of the art of the software for integer linear programming and hardware facilities of the present day computers may allow us to solve $M_2$. However, the combinatorial nature of the problem may not allow to find the Frobenius number of $a_1, a_2, \dots, a_n$ when $a_1$ becomes larger and larger. Therefore, we need more user friendly models that require less CPU time of the optimizer that is used for their solutions.

## III. RESIDUE APPROACH

Let $\alpha$ be the Frobenius number of $a_1, a_2, \dots, a_n$. In what follows we drop the subscript 1 from $a_1$ and put $a_1 = a$. Consider the partition of the set of nonnegative integers into residue classes modulo $a$. Let $C_0,\ C_1, \dots, C_{a-1}$ be the residue classes repre-sented, respectively, by $0, 1, \dots, a-1$; that is

$$C_j = \{ka + j : k \geq 0\ and\ integer\}, j = 0, 1, ., a-1$$

We note that $\alpha$ belongs to one of these residue classes. Note also that $\alpha$ can not belong to $C_0$, because otherwise $\alpha$ would be a multiple of $a = a_1$, hence a NILC of $a_1, a_2, \dots, a_n$. So, $\alpha$ must belong to one of the classes $C_1$, $C_2, \dots, C_{a-1}$. Now for each $j = 1, \dots, a-1$, let $\alpha_j$ be the smallest element that can be expressed as a NILC of $a_1, a_2, \dots, a_n$. Say

$\alpha_j = k_j a + j$. Then $\alpha_j - a$ can not be expressed as a NILC of $a_1, a_2, \dots, a_n$, but any element $\beta_j$ of $C_j$ with $\beta_j \geq \alpha_j$ can be expressed as such. In fact, we have $\beta_j = h_j a + j$ with $h_j \geq k_j$ so that

$$\beta_j = h_j a + j = k_j a + j + (h_j - k_j)a = \alpha_j + (h_j - k_j)a$$

is a NILC of $a_1, a_2, \dots, a_n$.

The above discussion shows that (see also, Brauer and Shockley [9]),

$$\alpha = \max\{\alpha_1 - a, \alpha_2 - a, \dots, \alpha_{a-1} - a\} \quad (5)$$

Various special algorithms are proposed for obtaining $\alpha_1, \alpha_2, \dots, \alpha_{a-1}$. (see for instance, Owens [19]; Beihoffer et al [6]; Böcker and Liptak [7]). We propose the following model for finding $\alpha_j$ for each $j = 1, \dots, a-1$.

$M_3(j)$:

$$\begin{aligned} &\min \alpha_j \\ &s.t. \\ &a_2 x_2 + \cdots + a_n x_n - a y_j = j \\ &\alpha_j - a y_j = j \\ &x_i \geq 0, y_j \geq 0, \alpha_j \geq 0 \text{ and integer}, i = 2, \dots, n \end{aligned}$$

For each $j = 1, \dots, a-1$, this formulation has two constraints and $n+2$ decision variables. By writing a computer code that uses any integer programming solver, like CPLEX, as a subroutine, one can obtain the numbers $\alpha_1, \alpha_2, \dots, \alpha_{a-1}$ easily. Then one gets the Frobenius number by the relation given in (5).

**Example 3.1.** Let us use $M_3$ to find F(34, 37, 38, 40, 43). $M_3(j)$ becomes:

$$\begin{aligned} &\min \alpha_j \\ &s.t. \\ &37x_2 + 38x_3 + 40x_4 + 43x_5 - 34y_j = j \\ &\alpha_j - 34y_j = j \\ &x_i \geq 0, y_j \geq 0, \alpha_j \geq 0 \text{ and integer}, i = 2, 3, 4, \end{aligned}$$

for each $j = 1, \dots, 33$. Optimal solution of the above model for each $j$ is given in Table 1 below.

| $j$ | $y_j$ | $\alpha_j$ | $j$ | $y_j$ | $\alpha_j$ | $j$ | $y_j$ | $\alpha_j$ |
|---|---|---|---|---|---|---|---|---|
| 1 | 6 | 205 | 2 | 5 | 172 | 3 | 1 | 37 |
| 4 | 1 | 38 | **5** | **6** | **209** | 6 | 1 | 40 |
| 7 | 2 | 75 | 8 | 2 | 76 | 9 | 1 | 43 |
| 10 | 2 | 78 | 11 | 3 | 113 | 12 | 2 | 80 |
| 13 | 2 | 81 | 14 | 3 | 116 | 15 | 2 | 83 |
| 16 | 3 | 118 | 17 | 3 | 119 | 18 | 2 | 86 |
| 19 | 3 | 121 | 20 | 4 | 156 | 21 | 3 | 123 |
| 22 | 3 | 124 | 23 | 4 | 159 | 24 | 3 | 126 |
| 25 | 4 | 161 | 26 | 4 | 162 | 27 | 3 | 129 |
| 28 | 4 | 164 | 29 | 5 | 199 | 30 | 4 | 166 |
| 31 | 4 | 167 | 32 | 5 | 202 | 33 | 4 | 169 |

**Table 1.** Optimal solutions of $M_3(j)$, $1 \leq j \leq 33$

We see from Table 1 that

$$\max\{\alpha_j : 1 \le j \le 33\} = 209.$$

So F(34, 37, 38, 40, 43) = 209 − 34 = 175. The mean CPU of the 33 cases has been found as 0.03 seconds.

Note that if $\alpha_j = ay_j + j$ and $y = \max\{\alpha_j : 1 \le j \le a - 1\}$, then $y - a = F(a_1, a_2, \dots, a_n)$ and $ay_j + j \le y$ for all $j = 1, \dots, a - 1$.

In order to avoid solving $a - 1$ models separately or to prepare a special code, we propose the following model for finding the FN directly.

$M_4$:

$$\min F$$
$$s.t.$$
$$a_2x_{2j} + \cdots + a_nx_{nj} - ay_j = j, j = 1, \dots, a - 1$$
$$y - ay_j \ge j, j = 1, \dots, a - 1$$
$$y - F = a$$
$$x_{ij} \ge 0, y_j \ge 0, F \ge 0 \text{ and integers}, \quad i = 2, \dots, n, \; j = 1, \dots, a - 1$$

If the value of F in the optimal solution of $M_4$ is $\bar{F}$ then F($a_1, a_2, \dots, a_n$) = $\bar{F}$. The formulation $M_4$ has $n(a - 1) + 2$ integer decision variables and $2a - 1$ contstraints. The dimension of $M_4$ increases very rapidly depending upon the value of $a = a_1$.

Therefore, when the solution time is important, the researcher may prefer the model $M_3$.

**Example 3.2.** Frobenius numbers of various combinations of positive integers have been calculated by using $M_4$. Our findings are given in Table 2 below. CPU time of each instance has been

| PIs | FN |
|---|---|
| 10, 195, 218 | 1057 |
| 10, 195, 218, 272 | 729 |
| 10, 195, 218, 272, 287 | 621 |
| 10, 195, 218, 272, 287, 324 | 601 |
| 10, 195, 218, 272, 287, 324, 341 | 509 |
| 10, 195, 218, 272, 287, 324, 341, 353, 499 | 489 |
| 11, 882, 1693, 2749, 3727 | 5210 |
| 11, 893, 1017 | 3809 |
| 11, 893, 1017, 1217 | 3440 |

**Table 2.** Frobenius Number of various sets of PIs

| PIs | FN | Time(sec) |
|---|---|---|
| 5123, 5692, 6055 | 972404 | 685.045 |
| 5123, 5692, 6055, 6371 | 267783 | 353.396 |
| 5123, 5692, 6055, 6371,6899 | 150698 | 386.683 |
| 5123, 5692, 6055, 6371,6899,7300 | 106857 | 580.19 |
| 5123, 5692, 6055, 6371,6899,7300,8472 | 85227 | 676.356 |
| 5123, 5692, 6055, 6371,6899,7300,8472,8619 | 72179 | 794.273 |
| 5123, 5692, 6055, 6371,6899,7300,8472,8619,9001 | 67678 | 925.803 |
| 5123, 5692, 6055, 6371,6899,7300,8472,8619,9001,9544 | 60851 | 1002.52 |
| 5123, 5692, 6055, 6371,6899,7300,8472,8619,9001,9544, 9809 | 56274 | 1113.33 |
| 5123, 5692, 6055, 6371,6899,7300,8472,8619,9001,9544, 9809,10012 | 54921 | 1219.73 |
| 5123, 5692, 6055, 6371,6899,7300,8472,8619,9001,9544, 9809,10012,11207 | 51648 | 1316.25 |

**Table 3.** Frobenius Number of various sets with larger $a_1s$

seconds. This model allows us to compute the Fro-
less than 5 seconds. The model allows us to compute the Frobenius number of a given set of positive integers $a_1 < a_2 < \cdots < a_n$, very easily, for small values of $a_1$.

We observe from Table 2 that, as we expect, there is
An inverse relation between the FN and the number of elements of the given set of PIs.

The dimension of the model $M_4$ depends merely upon $a_1$. Therefore, as it is seen in Table 2, it has been convenient to choose sets of integers with small $a_1$s.

Our computational analysis showed that, CPU time rapidly increases as $a_1$ gets larger. In order to reduce the computation time, we prepared a code with C++ CPLEX Concert Technology which solves $M_3$ for each $j$ by using CPLEX 12.8 as a subroutine and finally produces the FN of the given set of integers. Our code is able to reach the FN of a given set within seconds.

Computational results of some sequences with $a_1$s at levels of several thousands are given in Table 3.

## IV. CONCLUSION

We presented four integer linear programming formulations $(M_1, M_2, M_3, M_4)$ regarding the Frobenius problem. The formulations $M_1$ and $M_2$ can be used to get necessary and sufficient conditions for a positive integer to be the Frobenius number of a given set of relatively prime positive

integers. The formulations $M_3$ and $M_4$ are used to find the Frobenius number of a given set of relatively prime positive integers directly. Several examples are worked out with each formulation for demonstrative purposes.

The dimension of the proposed formulations are closely related with the smallest integer $a_1$ in the given set. To get $M_3$ work more effectively for larger $a_1$ s, we wrote a computer code which uses our $M_3$ with CPLEX 12.8; with that code we get the ability to compute the FN of any set of integers within seconds.

Our approach in this work is completely different from the existing algorithmic approaches. Anyone who wants to compute the Frobenius number of a given set of relatively prime positive integers can use our formulations easily. There are free optimizers in the internet, while the existing algorithms need special computer codes which can not be reached easily.

We hope that, this paper will open a new window for the computation of the Frobenius number by using integer programming techniques.

Adaptations of the proposed formulations for special classes of PIs are under construction.

## REFERENCES:


[1] Aardal K., Lenstra J.K., Hard Equality Constrained Integer Knapsacks, *Mathematics of Operations Research*, Vol.29, No.3, 2004, pp. 724-738.

[2] Alfonsin J.L.R., *The Diophantine Frobenius Problem*, Oxford Lecture Series in Mathematics and its applications, Vol.30, 1995.

[3] Alfonsin J.L.R., Complexity of the Frobenius Problem, *Combinatorica*, Vol.16, No.11, 1996, pp. 143-147.

[4] Aliev I., Henk M., Integer Knapsacks: Average Behavior of the Frobenius Numbers, *Mathematics of Operations Research*, Vol.34, No.3, 2009, pp. 698-705.

[5] Aliev I., Henk M., LLL-reduction for integer knapsacks, *Journal of Combinatorial Optimization*, Vol.24, No.4, 2012, pp. 613-626.

[6] Beihoffer D., Hendry J., Nijenhuis A., Wagon S., Faster Algorithms for Frobenius Numbers, *The electronic Journal of Combinatorics*, Vol.12, 2005.

[7] Böcker S., Liptak Z., A Fast and Simple Algorithm for the Money Changing Problem, *Algorithmica*, Vol.48, 2007, pp. 413-432.

[8] Brauer A., On a problem of partitions, *Amer J. Math*, Vol.64, 1942, pp. 299-312.

[9] Brauer A., Shockley J.E., On a problem of Frobenius, *J. reine angew. Math*, Vol.211, 1962, pp. 215-200.

[10] Curtis F., On formulas for the Frobenius number of a numerical semigroup, *Math Scand*, Vol.67, 1990, pp. 190-192.

[11] Delgado M., Garcia-Sanchez P.A., Morais J., *"NumericalSgps", A GAP Package for Numerical Semigroups Version 1.0.1*. 2015, http://www.gap-system.org/

[12] Einstein D., Lichtblau D., Strzebonski A., Wagon S., Frobenius numbers by Lattice point Enumeration, *Integers: Electronic Journal of Combinatorial Number Theory*, 2007.

[13] Hansen P., Ryan J., Testing integer knapsacks for feasibility, *European Journal of Ope-rational Research*, Vol.88, 1996, pp. 578-582.

[14] Krawczyk H., Paz A., The Diophantine problem of Frobenius: A close Bound, *Discrete Applied Mathematics*, Vol.23, 1989, pp. 289-291.

[15] Lewin M., An algorithm for a solution of a problem of Frobenius, *J. reine angew. Math*, Vol.276, 1975, pp. 68-82.

[16] Mohammed D.H., *Generalized Frobenius numbers: geometry of upper bounds, Frobenius graphs and exact formulas for arithmetic sequences*, Dissertation, Cardiff University, 2017.

[17] Navarro A.S.R., *Linear Diophantine equations and applications*, Dissertation, Universidad de Granada. 2015.

[18] Ong D.C., Ponomarenko V., The Frobenius Number of Geometric Sequences. *Integers: Electronic Journal of Combinatorial Number Theory*, Vol.8, No.1, 2008.

[19] Owens R.W., An Algorithm to solve the Frobenius Problem, *Mathematics Magazine*, Vol.76, No.4, 2003, pp. 264-275.

[20] Roberts J.B., Note on Linear forms, *Proc. Amer. Math. Soc.*, Vol.7, 1956, pp. 465-469.

[21] Roune B.H., Solving thousand-digit Frobenius Problem using Gröbner bases, *Journal of Symbolic Computation*, Vol.43, 2008, pp. 1-7.

[22] Ryan, J., The structure of an ıntegral monoid and Integer Programming feasibility, *Discrete Applied Mathematics*, Vol.28, 1990, pp. 251-263.

[23] Sylvester J.J., Mathematical questions with their solutions. *Educational Times*, Vol.41, 1884, 21.

[24] The On - Line Encyclopedia of Integer Sequences, http://oeis.org

[25] Trimm J.E., *On Frobenius numbers in three variables*, Dissertation, Auburn University – Alabama. 2006.

[26] Tripathi A., Formulae for the Frobenius number in three variables, *Journal of Number Theory*, Vol.170, 2017, pp. 368-389.

[27] Vizvari B., An application of an optimal behaviour of the greedy solution in number theory, *Periodica Mathematica Hungarica*, Vol.27, No.2, 1993, pp. 69-83.