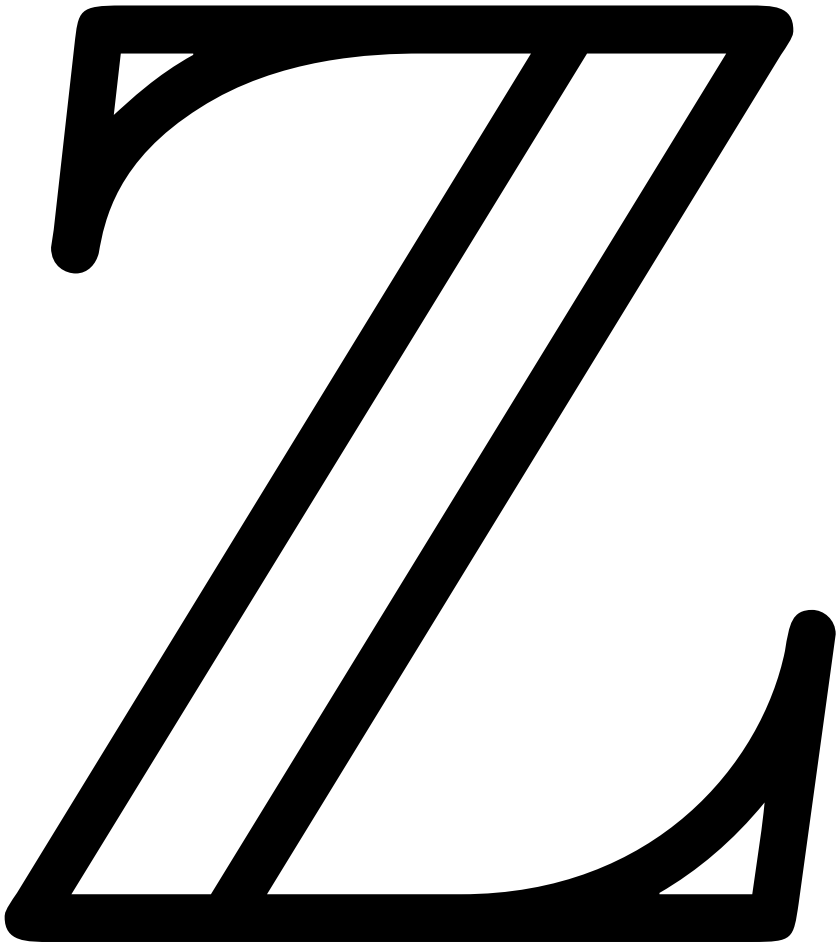
Katedralskolan, Uppsala

Finding algorithms to identify integers in a binary matrix given row and column sums

By Artem Los



Candidate number: 1291-0
Supervisor: Mauritz Blomqvist
Word count: 3900

Abstract

In this essay binary matrices are going to be examined from the observed symmetries, with focus on matrices with the same row sums and column sums configuration (here called isomorphs). The main problem of the essay is to find a way to convert between the matrix and the row sums and column sums configuration, hence the question:

Finding algorithms to identify integers in a binary matrix given row and column sums.

In the beginning of the essay, some of the properties of isomorphs are going to be described, in order to explain the symmetry found in the numerical difference of isomorphs. Later on, the relation between the size of a matrix and the number of isomorphs is going to be found to be similar to an already existing closed form in the Encyclopædia of Integer Sequences, which appears to be the solution for this problem. Given the closed form for the number of isomorphs in a matrix with a specific size($isomorphs$), and the maximum value that can be stored inside that matrix(max), it is going to be found that:

$$\frac{isomorphs}{max} \to 1$$

This fact implies that the smaller size a matrix has, the more values can be stored inside and later retrieved using the row sums and column sums configuration. Finally, the algorithm for identification of isomorphs is going to be presented. It will show that by simplifying the row sums and column sums configuration that is unknown, it is possible to obtain a matrix ID that is already known. In the identification process, either the original matrix is going to be found or the number of possible solutions. In the discussion, alternative interpretation using systems of equations and encryption using isomorphs are going to be presented, which are the practical usages of the theorems and the algorithms found during this research.

Word count: 299

Table of Contents

Contents

List of Tables

Introduction

Research question Finding algorithms to identify integers in a binary matrix given row and column sums.

Purpose The aim of this work is to examine the relation that exists between binary matrices and their row and column sums, in order to investigate if it is possible to switch between these forms, and so find algorithms that would facilitate the conversion process. In this work two different approaches are used. The first approach to the analysis is to investigate the numerical relation of matrices, while the second approach is to work with the symmetries that are going to be observed.

Definitions

Isomorph a matrix that shares the same row sums and column sums configuration with another matrix (these two matrices are called isomorphs). The matrices can be of different sizes. If row sums and column sums can be used to form more than one matrix (i.e. it has two or more solutions), it is an isomorph.

Isomorph in its simplest state a matrix that shares row sums and column sums configuration with only one matrix. The second matrix can be found by using transposition (see below). All other isomorphs that do not belong to this group are called non-simple isomorphs or isomorphs in non-simplest state.

Transposition when an isomorph is transposed, it is mirrored, given it is in the simplest state. For example, by transposing:

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

we get:

$$\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

Shape difference function a function that takes two parameters (in matrix form) and returns the smallest no. of moves of 1s needed to construct the second matrix. A move is the no. of times we need to change 1's position in either horizontal or vertical direction. Removing or adding a 1 counts as a move also. Defined as $d_i()$.

Matrix configuration notation a notation that describes the structure of a matrix using the no. of rows, no. of columns and the restricted set of allowed integers. For example, $z[m, n]$ where $z, m, n \in \mathbb{Z}^+$ is equal to:

$$\begin{bmatrix} x_1 & x_2 & \dots & x_n \\ \vdots & \vdots & \ddots & \\ x_{(m-1)n+1} & x_{(m-1)n+2} & \dots & x_{mn} \end{bmatrix}$$

where z gives the largest value that can be stored in one cell, e.g. $0 \leqslant x_i < z$. A matrix with the configuration $2[3, 2]$ would refer to matrices with three rows, and two columns, for instance:

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 1 \end{bmatrix}$$

The z value, which is 2, shows that only ones and zeroes can be stored in each cell. When $z = 2$, the matrix can be referred to as a binary matrix.

Matrix identification notation a specific notation used to express a specific matrix where the sum of row sums and the sum of column sums is known. In order to construct an ID, each row sum from top to bottom and finally each column from left to right. Referred to as *matrix ID*. Given a matrix 2[m,n], the matrix ID is in form of: $|R_1R_2 \dots R_m, C_1C_2 \dots C_n|$, where R_i is the sum of row i and C_i is the sum of column i.

Isomorphic function 1 a function that takes in one parameter (in matrix configuration notation) and returns the no. of isomorphs for that matrix configuration. Defined as $n_1()$.

Isomorphic function 2 a function that takes in one parameter (in matrix ID form) and returns the no. of isomorphs that can be formed using that matrix ID. Defined as $n_2()$.

Binary matrix A matrix that has a matrix configuration $2[m, n]$.

Theorems

Transposition of isomorphs in simplest state given an isomorph in its simplest state, the shape difference is always equal to 2.

Proof

$$d_i\left(\begin{bmatrix}0 & 1\\ 1 & 0\end{bmatrix}, \begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix}\right) = 2$$

□

Splitting up a non-simple isomorph into components Given an isomorph in the non-simplest state, it is always possible to split it up into components (smaller matrices, by splitting the original matrix in either vertical or horizontal direction), where at least one component is an isomorph in its simplest state.

Proof By definition, an isomorph has a pair, while the configuration of row sums and column sums remains unchanged. This means that if we have an isomorph which is not in the simplest state, its matrix ID will at least have two solutions, and these two unique solutions are caused by an isomorph in its simplest state. Also, as an example, let us start with an isomorph. If

$$\begin{bmatrix}0 & 1\\ 1 & 0\end{bmatrix}$$

is extended

$$\begin{bmatrix}0 & 1 & 0\\ 1 & 0 & 1\end{bmatrix}$$

So, the components in this case are

$$\begin{bmatrix}0 & 1\\ 1 & 0\end{bmatrix}$$

and

$$\begin{bmatrix}0\\ 1\end{bmatrix}$$

This will produce an isomorph, because we can transpose the simplest state isomorph, hence

$$\begin{bmatrix}1 & 0 & 0\\ 0 & 1 & 1\end{bmatrix}$$

Another way to split up an isomorph into components is as following.

$$\begin{bmatrix}1 & 1 & 0\\ 0 & 0 & 1\end{bmatrix} = \begin{bmatrix}1 & 0 & 0\\ 0 & 0 & 1\end{bmatrix} + \begin{bmatrix}0 & 1 & 0\\ 0 & 0 & 0\end{bmatrix}$$

where the isomorph component to the right of the equal sign is in its simplest state. □

Isomorph construction theorem Given an isomorph, it is always possible to construct a new isomorph by transposition of a simplest state component of that isomorph.

Proof In this proof two kinds of isomorphs are discussed separately. The first kind is an *isomorph in its simplest state*, and the second kind is not in its simplest state. These are all caused by the isomorph in its simplest state. In the simplest state, given that matrix ID is $|11, 11|$, i.e.

$$\begin{bmatrix}0 & 1\\ 1 & 0\end{bmatrix}$$

using transposition, a new isomorph is formed, i.e.

$$\begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix}$$

The new matrix has the same row sums and column sums as the first matrix, hence it is an isomorph. In the non-simplest state, we need to split it up into components. Using *splitting up a non-simple isomorph into components* theorem, we can construct new isomorphs using the *isomorph in its simplest state.* □

Numerical relation in isomorphs (in a matrix with constant matrix configuration) During this research, an interesting relation between isomorphs with constant matrix configuration and between different matrix configurations was found. This allows us to find isomorphs and non-isomoprhs numerically. The numerical value of a matrix is defined as the sum of every element in the matrix.

$$\begin{bmatrix} x_1 \times 2^0 & x_2 \times 2^1 & \dots & x_n \times 2^{n-1} \\ \vdots & \vdots & \ddots & \vdots \\ x_{(m-1)n+1} \times 2^{(m-1)n} & x_{(m-1)n+2} \times 2^{(m-1)n+1} & \dots & x_{mn} \times 2^{mn-1} \end{bmatrix}$$

The maximum value that can be stored in the matrix is a geometric progression.

$$\sum_{m,n\in\mathbb{Z}^+} 2^{mn-1} = \frac{2^{mn}-1}{2-1} = 2^{mn}-1$$

In the table below, a symmetrical structure can be observed. This is very important because we only need to know the numerical value of one less than a half of the isomorphs (as the middle value tends to be 3) in a specific matrix configuration in order to get the remaining ones. This symmetry was observed in matrices $2[3,2]$, $2[4,2]$, $2[5,2]$, $2[6,2]$, $2[7,2]$ and $2[8,2]$. The larger matrices could not be analysed in MS-Excel due to the big amount of data. The symmetrical structure can be explained by referring to the

Table 1: Numerical values of isomorphs in $2[3,2]$ in ascending order, including the computed numerical difference

Numerical value	Numerical difference
6	
9	3
18	9
22	4
24	2
25	1
26	1
27	1
30	3
33	3
36	3
37	1
38	1
39	1
41	2
45	4
54	9
57	3

definition of an isomorph. According to *Isomorph construction* theorem, given that one component is an isomorph it is possible to generate new isomorphs.

Now, by observing the shape difference of the pairs $\{6, 9\}$ and $\{54, 57\}$ we see that:

$$d_i\left(\begin{bmatrix}0 & 1\\ 1 & 0\\ 0 & 0\end{bmatrix}, \begin{bmatrix}1 & 0\\ 0 & 1\\ 0 & 0\end{bmatrix}\right) = d_i\left(\begin{bmatrix}0 & 1\\ 1 & 0\\ 1 & 1\end{bmatrix}, \begin{bmatrix}1 & 0\\ 0 & 1\\ 1 & 1\end{bmatrix}\right) = 2$$

It is the same kind of transposition going on, but in comparison to the simplest state isomorph transposition to the left, the right ones have two values in the last row, which makes them have the biggest numerical value in $2[3, 2]$.

There is a special pair of isomorphs that is in the middle, which results a numerical value and a shape difference of 3. It is special because as it is in the middle, it does not have another pair with similar numerical difference and shape difference. That is $\{30, 33\}$.

$$d_i\left(\begin{bmatrix}0 & 1\\ 1 & 1\\ 1 & 0\end{bmatrix}, \begin{bmatrix}1 & 0\\ 0 & 0\\ 0 & 1\end{bmatrix}\right) = 3$$

If we ignore the last row again, we can see that the matrix with numerical value of 30 cannot form more isomorphs by increasing its value in ascending order. If we add another value in x_1, the matrix will not be able to form isomorphs. Therefore, the last row is increased by one and we finally get 33, which is a numerical value of an isomorph in its simplest state. □

The relation in the no. of isomorphs (in a matrix with different matrix configuration) The previous section dealt with the numerical values of isomorphs, however, now we are to investigate how the no. of isomorphs varies between binary matrices with two rows, and compare it to the maximum no. of values that can be stored in that matrix.

It is important to keep in mind that isomorphs from $2[2, 2]$ can be expressed in $2[3, 2]$, when one row sum is zero, i.e. isomorphs of $2[2, 2] \in 2[3, 2]$ (we will discuss this a bit further in *Simplification*).

After some research using the Encyclopædia of Integer Sequences, we can quickly figure out that the closed form[1] of the sequence in *Table 2* might be:

$$n_1(2[n, 2]) = 4^n - 2 \times 3^n + 2^n \tag{1}$$

[1] A038721 - OEIS. 2013. A038721 - OEIS. [ONLINE] Available at: http://oeis.org/A038721/internal. [Accessed 03 August 2013].

Table 2: The total no. of isomorphs in a matrix with a specific matrix configuration. The values are computed using an algorithm, see Appendix 1.

Matrix configuration	No. of isomorphs
2[2,2]	2
2[3,2]	18
2[4,2]	110
2[5,2]	570
2[6,2]	2702
2[7,2]	12138
2[8,2]	52670
2[9,2]	223290

The interesting fact that can be observed is that the no. of isomorphs relative to the maximum value in a matrix approaches 1. For example,

$$\frac{2}{15} = 0.133\ldots, \frac{18}{63} = 0.286\ldots, \frac{110}{255} = 0.431\ldots$$

$$\frac{570}{1023} = 0.557\ldots, \frac{2702}{4095} = 0.660\ldots, \frac{12138}{16383} = 0.741\ldots$$

$$\frac{52670}{65535} = 0.804\ldots, \frac{223290}{262143} = 0.852\ldots$$

In general,

$$\lim_{n\to\infty} \frac{n_1(2[n,2])}{2^{2n}-1} = \lim_{n\to\infty} \frac{4^n - 2\times 3^n + 2^n}{4^n - 1} = \lim_{n\to\infty} \frac{4^n(1 - \frac{2\times 3^n}{4^n} + \frac{1}{2^n})}{4^n(1-\frac{1}{4^n})} = \lim_{n\to\infty} \frac{1 - \frac{2\times 3^n}{4^n} + \frac{1}{2^n}}{1-\frac{1}{4^n}} = 1 \tag{2}$$

This means that as the value of n increases, the more isomorphs we get in comparison to the maximum value. In terms of data storage inside matrices, this means that in order to save as much space as possible (the more isomorphs, the less data can be stored), we need to keep the value of n as small as possible assuming that the closed form in (eq. 1) is true for $n \in \mathbb{Z}^+$. □

Interestingly, when $n \to \infty$, the no. of non-isomorphs will be negligible.

Sum of row sums and sum of column sums relation In order to validate if a given row and column sums forms a real matrix, it can be proven that the sum of the row sums is equal to the sum of the column sums, i.e.

$$\sum R_i = \sum C_i \tag{3}$$

Proof The simplest proof to (3) above is following: as each element will affect both the row and the column sums, it does not matter from what direction we observe the matrix, the sum of row sums and column sums will be equal to each other. If we have a $2[2, 2]$ matrix, i.e.

$$\begin{bmatrix} x_1 & x_2 \\ x_3 & x_4 \end{bmatrix}$$

using the associative property in arithmetic, following is derived:

$$\begin{aligned} &R_1 + R_2 = (x_1 + x_2) + (x_3 + x_4) \\ &C_1 + C_2 = (x_1 + x_3) + (x_2 + x_4) \\ &\therefore R_1 + R_2 = C_1 + C_2 \end{aligned}$$

As the associative property applies to sums as well, this numerical relation between the sum of row sums and the sum of column sums is true for any matrix, i.e. for matrix configuration $z[m, n]$. □

Algorithms for identification of isomorphs

We can refer to isomorphs as unsolvable matrices as given their matrix ID, we would not be able to figure out the original matrix.

In this section, a method for identification of isomorphs is going to be presented. Identification of an isomorph, in this case, is a way to either find how the original matrix looked like, given the matrix ID, or, if the matrix is unsolvable, find the number of possible ways a matrix can be constructed, given the matrix ID.

The approach to solve this problem consists of two main steps. The first step is to *simplify the matrix ID* and the second step is to *identify the solutions*. A matrix ID that results a unique solution and the one that does not can be simplified. However, when we are to identify the matrix i.e. find the unique solution or the number of solutions, two different approaches are required.

Simplification It is important to notice that matrices can be simplified, keeping their initial properties. We have to be careful though, as *initial properties* implies that the matrix ID will still be either solvable (containing only one unique solution) or unsolvable (no unique solutions, i.e. an isomorph), and we are still going to get the no. of solutions for unsolvable matrix ID, however the important point is to always keep in mind the original matrix, so that it is known what item in the original matrix corresponds to the simplified model.

The nature of the simplification process is, to some extent, the fact that a matrix with the configuration $2[x-1,c]$, where c is a constant, can always be expressed in $2[x,c]$, $2[x+1,c]$, and $2[x+k,c]$, where c,k are constants independent of each other, using any row sums value set to zero. Here, the constant value c means that there is a constant no. of columns (using *matrix configuration notation* definition) This also works if the no. of rows is constant, so, $2[c,x-1]$ can always be expressed in $2[c,x+k]$ because it does not matter what is named a 'row' and what is named a 'column', as long as these to concepts are separated. For example,

$$\begin{bmatrix} 1 & 1 \\ 0 & 1 \\ 0 & 0 \end{bmatrix}$$

where the matrix ID $|210,12|$. As it can be seen, removing the last row results a similar matrix.

$$\begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}$$

where the matrix ID is $|21, 12|$. This means that it is possible to work with the matrix ID directly, when performing the simplification. If the matrix ID would be the only information, it would be possible to figure out the original matrix. If it would result an isomorph, only the no. of possible matrices would be obtained (see *Identification*).

The elimination of rows with the sum zero is performed because the 'zero' indicates that there cannot be any values in that row. Similarly, if the sum of a row is equal to the maximum value of row sums in that matrix, all the elements should be present, i.e. that row in the binary matrix will be filled with numbers. Hence we can use a similar approach as with zero row sums, but because the row is now filled with values, we will need to decrease the column sums by a factor of one. For example, with a matrix ID $|211, 22|$, we get:

$$\begin{bmatrix} 1 & 1 \\ 0 & 1 \\ 1 & 0 \end{bmatrix}$$

Here it is possible to see that the first row is entirely filled with values. So, given the matrix ID, we can figure out that the first row is complete, hence it can be removed. Now, we are left with:

$$\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

The matrix ID for the matrix above is $|11, 11|$. After a minute of thought we realize that this is the *isomorph in its simplest state*, and according to the definition, it can only contain two solutions. It can also be seen that only one transposition can be performed hence two unique matrices are going to be formed. Using transposition, we get the second solution:

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

In other words, given $|\mathbf{2}11, \mathit{22}|$, we can remove the first maximum value (marked in bold), and later decrease the values after the comma (marked in italics).

A better illustration behind the principles of maximum values is:

$$|\underline{211}, \underline{22}|$$

$$\swarrow \qquad \searrow$$

max: 2 max: 3

Note, the order of row/column sums does not matter if we want to calculate the no. of solutions or deduce whether it is an isomorph or a non-isomorph, i.e. for *simplification*, $|211, 22|$ is equivalent to $|121, 22|$, et cetera, as long as rows sums are separated from column sums. It can be clearly seen that the 2 on the left side can be removed, hence the right side has to be decreased by one. However, the right side cannot be simplified as 2 is not a maximum in that region. It might be good to note the matrix configuration of this matrix ID, because it can hint us the maximum value. It is:

$$2[3, 2]$$

or

$$2[2, 3]$$

So, we can see that very complex (unknown for us) matrix ID can be simplified to a simpler matrix ID that we already know.

In conclusion, in a simplification of a matrix procedure, the zeros in row/column sums can be removed. If the row/column sum contains the maximum value, it can be removed also, provided that column/row sums are decreased by one.

Identification Once the matrix ID has been simplified, we are able to identify whether the matrix is an isomorph or a non-isomorph. Once this is known, we are going to figure out the original matrix, or the no. of solutions the matrix ID has. First, a matrix ID that is going to generate a non-isomorph should be chosen from *Table 5* (in appendix 1), in this case, $|21021|$. After *simplification*, this would result:

The dimension of the matrix is 2×3, thus we know the position of the 'comma'.

- $|210, 21|$ - adding comma to distinguish between column and row sums
- $|21, 21|$ - removing the zeros
- $|1, 1|$ - removing the maximum values

Once a matrix ID can be simplified to $|1, 1|$, we can be certain that this matrix is a non-isomorph, because the simplified matrix ID means that

there is only one cell, and that cell contains a value. With other words, the maximum value in the simplified matrix ID is 1, hence it must be a value in the cell. This matrix ID can be expressed as a binary matrix:

$$\begin{bmatrix} 1 & 1 \\ 1 & 0 \\ 0 & 0 \end{bmatrix}$$

In *Table 6* (see appendix 1), $|1111, 04|$ produces a non-isomorph. After giving this a thought, we realize that this does not turn out so nicely as the previous example. By performing the simplification, we get $|1111, 4|$. Now, instead of claiming that $|1, 1|$ is the only outcome, we can generalize this a bit further and say that every matrix that is solvable, i.e. a non-isomorph, will contain a matrix ID where there is only one row and/or column. So, a matrix ID should be of the form $2[1, n]$ or $2[n, 1]$, where n is a positive integer. This observation is obvious if we simply think about a queue of binary numbers, by knowing the column sums, we can always find the original row. The row sum is not needed.

$$\begin{bmatrix} 1 & 0 & 1 & 1 \end{bmatrix}$$

The tricky part comes when we are to deal with isomorphs. There are two different cases, so matrix ID can be split up further into two new groups: simple configuration and non-simple configuration. Let us start with some examples of matrix ID that are simply configured. First, $|2211, 33|$, when simplified, we get $|11, 11|$. This configuration will produce an isomorph in its simplest state, and as it is defined, it only contains one transposition, hence only 2 solutions. Secondly, $|1211, 32| \to |111, 21|$. According to *Table 6*, there should be 3 solutions. Thirdly, taking $|11121, 42| \to |1111, 31|$, and so, 4 solutions according to *Table 3*. We might start to see a pattern that to obtain 4 solutions, we took $3+1$ in the matrix ID, in order to get 3 solutions, we took $2 + 1$, et cetera. It might be suggested that the no. of solution is the sum of rows/columns, if the matrix ID is simply configured (i.e. matrix ID should be $|1 \dots 1, k1|$)).

$$n_2(|1 \dots 1, k1|) = k + 1, k \in \mathbb{Z}^+ \tag{4}$$

The proof for (4) is very simple. By starting from a low case, say $|31, 111|$, i.e.

$$\begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

As it can be seen, three transpositions can be made.

$$\begin{bmatrix} 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

So, there are 4 solutions(we have transposed the matrix 3 times, including the first matrix we started with). More generally, a matrix ID $|(n+1)1, 1...1|$, i.e.

$$\begin{bmatrix} 0 & 1 & \dots & 1 & 1 \\ 1 & 0 & \dots & 0 & 0 \end{bmatrix}$$

contains $n+1$ transpositions, hence the no. of solutions is: $n+1+1 = n+2$.

The more tricky part is left because we now have to deal with non-simple configured matrix ID. In *Table 6*, $|1111, 22|$ can be found. The no. of isomorphs in that matrix is 6, however, it cannot be deduced so easily. For this purpose, you need to look in the table on the next page.

In *Table 3* you can see the common matrix IDs and the no. of isomorphs they will produce. Please notice that the matrix ID has to be simplified before you can use this table.

The 'simp' means that there is also a simply configured ID that would produce the given amount of isomorphs. For instance, a non-simple configured matrix ID $|22, 1111|$ will produce six isomorphs, however, the 'simp' implies that this can be done using a simply configured matrix ID, $|51, 111111|$ as well. The 'three dots' means that the no. of ones '1' can be obtained using *Sum of row sums and sum of column sums* relation discussed earlier. For example, $|33, 1\dots|$ is actually $|33, 111111|$, or with other words, there are six ones, as $3 + 3 = 6$.

Table 3: Matrix ID and the corresponding no. of solutions. Computed using algorithm in Appendix 1

No. of solutions	Matrix ID
2	11,11
3	21,111
4	31,1111
5	41,11111
6	22,1111; (or simp)
7	61,1111111
8	71,11111111
9	81,111111111
10	32,11111 (or simp)
15	42,1 . . . (or simp)
20	33,1 . . . (or simp)
21	25,1 . . . (or simp)
28	26,1 . . . (or simp)
35	43,1 . . . (or simp)
36	72,1 . . . (or simp)
56	53,1 . . . (or simp)
70	44,1 . . . (or simp)
126	54,1 . . . (or simp)

Discussion

Improvements During this research I tried to examine some of the properties of binary matrices, with the aim to find a way to identify unsolvable matrices (isomorphs) and see if there is a relation between the size of the matrix and the no. of isomorphs. Because of the time limitation and the word count restriction, I was able to investigate binary matrices only (i.e. $2[m,n]$), although some of the theorems can be applied in other matrices as well. The greatest emphasis was put on the observed symmetries in binary matrices, which was discussed in *Algorithms for identification of isomorphs*. In my mind it is much easier to find isomorphs by working with the matrix ID, so less effort was put into the investigation of the numerical relation between matrices. With more time and more words, I would continue to find an even stronger proof for the numerical relations, include more facts that were not included in the essay, examine the relation in matrices with bigger set of possible integers, and analyse larger matrices.

Alternative interpretation The observed symmetries can be used to solve different kinds of problems. One of them is the solution to systems of equations, when it is given that the values are in a specific range. By translating the problem into a matrix ID, it might simplify the process of finding a solution or the no. of solutions.

$$\begin{bmatrix} x_1 & x_2 \\ x_3 & x_4 \end{bmatrix} \equiv |R_1R_2, C_1C_2| \equiv \begin{cases} R_1 = x_1 + x_2 \\ R_2 = x_3 + x_4 \\ C_1 = x_1 + x_3 \\ C_2 = x_2 + x_4 \end{cases}$$

For example, $|11, 11|$ has two solutions, so does the system of equations below. Either $x_1 = x_4 = 1$ and $x_2 = x_3 = 0$, or $x_1 = x_4 = 0$ and $x_2 = x_3 = 1$.

$$\begin{cases} 1 = x_1 + x_2 \\ 1 = x_3 + x_4 \\ 1 = x_1 + x_3 \\ 1 = x_2 + x_4 \end{cases}$$

Therefore, if we know how to solve the system of equations, we can find a way to solve these matrices, and vice versa.

Encryption using isomorphs One of the practical usages of this study about isomorphs is the protection of data. This cipher is going to be based on a $2[2,2]$ matrix, as it has the smallest amount of isomorphs (see the fifth theorem [2],(eq.3)).

In order to encrypt, first, we need an open message and a key, where every letter is expressed in radix 15. Secondly, we should add each value in the open message with the corresponding in the key. If the length of the key is less than length of the message, repeat the key. Thirdly, find the corresponding binary value in the table below.

Table 4: Binary value corresponding to a decimal value

0000	0
0001	1
0010	2
0011	3
0100	4
0101	5
0110	6
0111	7
1000	8
1010	9
1011	10
1100	11
1101	12
1110	13
1111	14

A matrix ID should be calculated based on the binary value. The matrix ID represents the number in radix 3. Now, we can omit a digit in the matrix ID, as it can be retrieved later using the sixth theorem[3]. Now, convert the rest of the matrix ID (three digits, ignoring one digit) into radix 10. The final numbers can be stored in binary with maximum 5 bits.

In order to decrypt, the encrypted message and the key are required. First, take the encrypted message and convert it to radix 3 (if it is in radix 2, convert it to radix 10 first.) Depending on which digit was omitted, it

[2]The relation in the no. of isomorphs (in a matrix with different matrix configuration) theorem

[3]Sum of row sums and sum of column sums relation theorem

can be found using the sixth theorem[4]. Now, identify (see *Identification*) the values inside the matrix. Use this information and the *Table 4* to retrieve the corresponding value in radix 15. If the matrix ID results an isomorph, assume it is 6. We have now deciphered the messaged.

[4]Sum of row sums and sum of column sums relation theorem

Conclusion

In this essay we have examined the relation between matrix IDs and the corresponding matrices in order to find a way to convert between them. It was found that when the matrix is a non-isomorph given the row and column sums (matrix ID), the original matrix can be found. However, if the matrix is an isomorph, given the row and column sums, the no. of possible solutions can be found. If the matrix ID (row and column sums) is simply configured, using either the row sums or column sums, the no. of solutions can be found directly. If the matrix ID is not simply configured, a specific table has to be used.

Bibliography

Sources

- A038721 - OEIS. 2013. A038721 - OEIS. [ONLINE] Available at: http://oeis.org/A038721/internal. [Accessed 03 August 2013].

Tools used

- This document is written in TEXstudio and compiled with PDFLaTeX.exe
- Raw data was converted to Microsoft Excel 2013, processed, and later converted to LATEXformat using Excel2LaTeX.
- Converter module in Mathos Laboratory (Mathos Project): http://labs.mathosproject.com/Default.aspx
- Basic calculations were performed using the mathematical expression parser Mathos Parser Web App (Mathos Project): http://parser.mathosproject.com/

Appendix 1 - List of calculated data using a computer simulation

In this appendix you will find results for different matrix IDs and matrix configuration notations.

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

using System.Threading;
using System.Threading.Tasks;

using System.IO;

namespace EE
{
    class Program
    {
        static void Main(string[] args)
        {
            int x = 2;  // b
            int y = 10;  // a               -> z[a,b] |2,0

            StreamWriter sw = new StreamWriter("c:\\EEResults\\
                key" + y + "x" + x + ".txt");

            List<string> ids = new List<string>();

            Dictionary<string, int> ids2 = new Dictionary<
                string, int>();

            for (int i = 0; i < CalculateMaxValue(x, y); i++)
            {
                int[,] matrix = ConvertNumberToMatrix(x, y, i);
                string id = CalculateID(matrix, x, y);
                Console.WriteLine("{0} - {1}", i, id);

                if (ids.Contains(id) == false)
                {
                    ids.Add(id);

                    ids2.Add(id, 1);

                }
```

```
39                  else
40                  {
41                      ids2[id] = ids2[id] + 1;
42
43                      ids.Add(id);
44
45                      sw.WriteLine(ids.IndexOf(id));
46                      sw.WriteLine(i);
47
48                  }
49              }
50
51              sw.Close();
52
53              StreamWriter sw2 = new StreamWriter("c:\\EEREsults
                    \\keys" + y + "x" + x + "N.txt");
54
55              foreach (var item in ids2)
56              {
57                  sw2.WriteLine("ID:␣" + item.Key  + ";␣N:␣" +
                        item.Value);
58              }
59
60              sw2.Close();
61              Console.Read();
62
63          }
64
65          static int CalculateMaxValue(int x, int y)
66          {
67              return (int)Math.Pow(2,x*y) ;
68          }
69          static string CalculateID(int[,] matrix,int x, int y)
70          {
71              string id = "";
72              for (int i = 0; i < y; i++)
73              {
74                  int xSum = 0;
75                  for (int j = 0; j < x; j++)
76                  {
77                      xSum += matrix[j,i];
78                  }
79                  id += xSum.ToString();
80              }
81
82              for (int i = 0; i < x; i++)
83              {
84                  int ySum = 0;
85                  for (int j = 0; j < y; j++)
```

```
86                  {
87                      ySum += matrix[i, j];
88                  }
89                  id += ySum.ToString();
90              }
91              return id;
92          }
93          static int[,] ConvertNumberToMatrix(int x, int y, int s
                )
94          {
95              int[,] matrix = new int[x,y];
96              string num = base10ToBase2B(s);
97
98              int k = 0;
99              for (int i =0; i < y; i++)
100             {
101                 for (int j = 0; j < x; j++)
102                 {
103
104                     matrix[j, i] = (int)num[k] == 49 ? 1 : 0;
105                     k++;
106                     if (k >= num.Length)
107                     {
108                         return matrix;
109                     }
110                 }
111             }
112
113
114             return matrix;
115         }
116
117         static string base10ToBase2B(int s)
118         {
119             // This code is based on the code snippet written
                    by Artem Los
120             // http://skgl.codeplex.com/
121             // License: http://skgl.codeplex.com/license
122
123             // This method is converting a base 10 number to
                    base 26 number.
124             // Remember that s is a decimal, and the size is
                    limited.
125             // In order to get size, type Decimal.MaxValue.
126             //
127             // Note that this method will still work, even
                    though you only
128             // can add, subtract numbers in range of 15 digits.
129             char[] allowedLetters = "01".ToCharArray();
```

```
130
131             int num = s;
132             int reminder = 0;
133
134             char[] result = new char[1024];
135             int j = 0;
136
137
138             while ((num >= 2))
139             {
140                 reminder = num % 2;
141                 result[j] = allowedLetters[reminder];
142                 num = (num - reminder) / 2;
143                 j += 1;
144             }
145
146             result[j] = allowedLetters[num];
147
148             return new string(result);
149
150         }
151         static int Base10ToBas2e(int s)
152         {
153             return Convert.ToInt32(Convert.ToString(s, 2));
154         }
155
156         static int mod(int n, int b)
157         {
158             return n - b * (int)Math.Floor((double)n / b);
159         }
160         static int q(int n, int b)
161         {
162             return b * (int)Math.Floor((double)n / b);
163         }
164
165     }
166 }
```

Algorithm for generation of isomorphs

```
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Text;
5
6  namespace EE
7  {
8      class Program
9      {
```

```
10          static void Main(string[] args)
11          {
12              // This method can be used to increase the
                    probability that the chosen number will be an
                    isomorph:
13              // The code works on the principle that if a ^ b =
                    b - a, b and are more likely to be isomorphs.
14
15              Random r = new Random();
16              for (int i = 0; i < 100; i++)
17              {
18
19                  int a = Convert.ToInt32( r.NextDouble() * 99) ;
20                  int b = Convert.ToInt32(r.NextDouble() * 99);
21
22                  if ( ( (((a ^ b) == b - a) && b > a) || (((a ^
                        b) == a - b) && a > b) ) && a>5 && b > 5  )
23                  {
24                      Console.WriteLine(a + "␣and␣" + b);
25                  }
26              }
27
28              Console.WriteLine("Done");
29              Console.ReadLine();
30          }
31      }
32  }
```

Table 5: No. of solutions for matrix IDs in $2[3,2]$

ID: 00000; N: 1	ID: 00101; N: 1
ID: 10010; N: 1	ID: 10102; N: 1
ID: 10001; N: 1	ID: 20112; N: 1
ID: 20011; N: 1	ID: 01102; N: 1
ID: 01010; N: 1	ID: 11103; N: 1
ID: 11020; N: 1	ID: 21113; N: 1
ID: 11011; N: 2	ID: 02112; N: 1
ID: 21021; N: 1	ID: 12113; N: 1
ID: 01001; N: 1	ID: 22123; N: 1
ID: 11002; N: 1	ID: 00211; N: 1
ID: 21012; N: 1	ID: 10221; N: 1
ID: 02011; N: 1	ID: 10212; N: 1
ID: 12021; N: 1	ID: 20222; N: 1
ID: 12012; N: 1	ID: 01221; N: 1
ID: 22022; N: 1	ID: 11231; N: 1
ID: 00110; N: 1	ID: 11222; N: 2
ID: 10120; N: 1	ID: 21232; N: 1
ID: 10111; N: 2	ID: 01212; N: 1
ID: 20121; N: 1	ID: 11213; N: 1
ID: 01120; N: 1	ID: 21223; N: 1
ID: 11130; N: 1	ID: 02222; N: 1
ID: 11121; N: 3	ID: 12232; N: 1
ID: 21131; N: 1	ID: 12223; N: 1
ID: 01111; N: 2	ID: 22233; N: 1
ID: 11112; N: 3	
ID: 21122; N: 2	
ID: 02121; N: 1	
ID: 12131; N: 1	
ID: 12122; N: 2	
ID: 22132; N: 1	

Table 6: No. of solutions for matrix IDs in $2[4,2]$

ID: 000000; N: 1	ID: 221023; N: 1	ID: 211141; N: 1	ID: 020112; N: 1	ID: 101222; N: 2
ID: 100010; N: 1	ID: 002011; N: 1	ID: 011121; N: 3	ID: 120113; N: 1	ID: 201232; N: 1
ID: 100001; N: 1	ID: 102021; N: 1	ID: 111122; N: 6	ID: 220123; N: 1	ID: 011231; N: 1
ID: 200011; N: 1	ID: 102012; N: 1	ID: 211132; N: 3	ID: 001102; N: 1	ID: 111241; N: 1
ID: 010010; N: 1	ID: 202022; N: 1	ID: 021131; N: 1	ID: 101103; N: 1	ID: 111232; N: 3
ID: 110020; N: 1	ID: 012021; N: 1	ID: 121141; N: 1	ID: 201113; N: 1	ID: 211242; N: 1
ID: 110011; N: 2	ID: 112031; N: 1	ID: 121132; N: 3	ID: 011103; N: 1	ID: 011222; N: 2
ID: 210021; N: 1	ID: 112022; N: 2	ID: 221142; N: 1	ID: 111104; N: 1	ID: 111223; N: 3
ID: 010001; N: 1	ID: 212032; N: 1	ID: 001111; N: 2	ID: 211114; N: 1	ID: 211233; N: 2
ID: 110002; N: 1	ID: 012012; N: 1	ID: 101112; N: 3	ID: 021113; N: 1	ID: 021232; N: 1
ID: 210012; N: 1	ID: 112013; N: 1	ID: 201122; N: 2	ID: 121114; N: 1	ID: 121242; N: 1
ID: 020011; N: 1	ID: 212023; N: 1	ID: 011112; N: 3	ID: 221124; N: 1	ID: 121233; N: 2
ID: 120021; N: 1	ID: 022022; N: 1	ID: 111113; N: 4	ID: 002112; N: 1	ID: 221243; N: 1
ID: 120012; N: 1	ID: 122032; N: 1	ID: 211123; N: 3	ID: 102113; N: 1	ID: 001212; N: 1
ID: 220022; N: 1	ID: 122023; N: 1	ID: 021122; N: 2	ID: 202123; N: 1	ID: 101213; N: 1
ID: 001010; N: 1	ID: 222033; N: 1	ID: 121123; N: 3	ID: 012113; N: 1	ID: 201223; N: 1
ID: 101020; N: 1	ID: 000110; N: 1	ID: 221133; N: 2	ID: 112114; N: 1	ID: 011213; N: 1
ID: 101011; N: 2	ID: 100120; N: 1	ID: 002121; N: 1	ID: 212124; N: 1	ID: 111214; N: 1
ID: 201021; N: 1	ID: 100111; N: 2	ID: 102131; N: 1	ID: 022123; N: 1	ID: 211224; N: 1
ID: 011020; N: 1	ID: 200121; N: 1	ID: 102122; N: 2	ID: 122124; N: 1	ID: 021223; N: 1
ID: 111030; N: 1	ID: 010120; N: 1	ID: 202132; N: 1	ID: 222134; N: 1	ID: 121224; N: 1
ID: 111021; N: 3	ID: 110130; N: 1	ID: 012131; N: 1	ID: 000211; N: 1	ID: 221234; N: 1
ID: 211031; N: 1	ID: 110121; N: 3	ID: 112141; N: 1	ID: 100221; N: 1	ID: 002222; N: 1
ID: 011011; N: 2	ID: 210131; N: 1	ID: 112132; N: 3	ID: 100212; N: 1	ID: 102232; N: 1
ID: 111012; N: 3	ID: 010111; N: 2	ID: 212142; N: 1	ID: 200222; N: 1	ID: 102223; N: 1
ID: 211022; N: 2	ID: 110112; N: 3	ID: 012122; N: 2	ID: 010221; N: 1	ID: 202233; N: 1
ID: 021021; N: 1	ID: 210122; N: 2	ID: 112123; N: 3	ID: 110231; N: 1	ID: 012232; N: 1
ID: 121031; N: 1	ID: 020121; N: 1	ID: 212133; N: 2	ID: 110222; N: 2	ID: 112242; N: 1
ID: 121022; N: 2	ID: 120131; N: 1	ID: 022132; N: 1	ID: 210232; N: 1	ID: 112233; N: 2
ID: 221032; N: 1	ID: 120122; N: 2	ID: 122142; N: 1	ID: 010212; N: 1	ID: 212243; N: 1
ID: 001001; N: 1	ID: 220132; N: 1	ID: 122133; N: 2	ID: 110213; N: 1	ID: 012223; N: 1
ID: 101002; N: 1	ID: 001120; N: 1	ID: 222143; N: 1	ID: 210223; N: 1	ID: 112224; N: 1
ID: 201012; N: 1	ID: 101130; N: 1	ID: 000101; N: 1	ID: 020222; N: 1	ID: 212234; N: 1
ID: 011002; N: 1	ID: 101121; N: 3	ID: 100102; N: 1	ID: 120232; N: 1	ID: 022233; N: 1
ID: 111003; N: 1	ID: 201131; N: 1	ID: 200112; N: 1	ID: 120223; N: 1	ID: 122243; N: 1
ID: 211013; N: 1	ID: 011130; N: 1	ID: 010102; N: 1	ID: 220233; N: 1	ID: 122234; N: 1
ID: 021012; N: 1	ID: 111140; N: 1	ID: 110103; N: 1	ID: 001221; N: 1	ID: 222244; N: 1
ID: 121013; N: 1	ID: 111131; N: 4	ID: 210113; N: 1	ID: 101231; N: 1	

Table 7: No. of solutions for some matrix IDs in $2[5,2]$

ID: 1000120; N: 1	ID: 0020121; N: 1	ID: 0111131; N: 4	ID: 1201123; N: 3	ID: 2012142; N: 1
ID: 1000111; N: 2	ID: 1020131; N: 1	ID: 1111132; N: 10	ID: 2201133; N: 2	ID: 0112141; N: 1
ID: 2000121; N: 1	ID: 1020122; N: 2	ID: 2111142; N: 4	ID: 0011112; N: 3	ID: 1112151; N: 1
ID: 0100120; N: 1	ID: 2020132; N: 1	ID: 0211141; N: 1	ID: 1011113; N: 4	ID: 1112142; N: 4
ID: 1100130; N: 1	ID: 0120131; N: 1	ID: 1211151; N: 1	ID: 2011123; N: 3	ID: 2112152; N: 1
ID: 1100121; N: 3	ID: 1120141; N: 1	ID: 1211142; N: 4	ID: 0111113; N: 4	ID: 0112132; N: 3
ID: 2100131; N: 1	ID: 1120132; N: 3	ID: 2211152; N: 1	ID: 1111114; N: 5	ID: 1112133; N: 6
ID: 0100111; N: 2	ID: 2120142; N: 1	ID: 0011121; N: 3	ID: 2111124; N: 4	ID: 2112143; N: 3
ID: 1100112; N: 3	ID: 0120122; N: 2	ID: 1011122; N: 6	ID: 0211123; N: 3	ID: 0212142; N: 1
ID: 2100122; N: 2	ID: 1120123; N: 3	ID: 2011132; N: 3	ID: 1211124; N: 4	ID: 1212152; N: 1
ID: 0200121; N: 1	ID: 2120133; N: 2	ID: 0111122; N: 6	ID: 2211134; N: 3	ID: 1212143; N: 3
ID: 1200131; N: 1	ID: 0220132; N: 1	ID: 1111123; N: 10	ID: 0021122; N: 2	ID: 2212153; N: 1
ID: 1200122; N: 2	ID: 1220142; N: 1	ID: 2111133; N: 6	ID: 1021123; N: 3	ID: 0012122; N: 2
ID: 2200132; N: 1	ID: 1220133; N: 2	ID: 0211132; N: 3	ID: 2021133; N: 2	ID: 1012123; N: 3
ID: 0010120; N: 1	ID: 2220143; N: 1	ID: 1211133; N: 6	ID: 0121123; N: 3	ID: 2012133; N: 2
ID: 1010130; N: 1	ID: 0001120; N: 1	ID: 2211143; N: 3	ID: 1121124; N: 4	ID: 0112123; N: 3
ID: 1010121; N: 3	ID: 1001130; N: 1	ID: 0021131; N: 1	ID: 2121134; N: 3	ID: 1112124; N: 4
ID: 2010131; N: 1	ID: 1001121; N: 3	ID: 1021141; N: 1	ID: 0221133; N: 2	ID: 2112134; N: 3
ID: 0110130; N: 1	ID: 2001131; N: 1	ID: 1021132; N: 3	ID: 1221134; N: 3	ID: 0212133; N: 2
ID: 1110140; N: 1	ID: 0101130; N: 1	ID: 2021142; N: 1	ID: 2221144; N: 2	ID: 1212134; N: 3
ID: 1110131; N: 4	ID: 1101140; N: 1	ID: 0121141; N: 1	ID: 0002121; N: 1	ID: 2212144; N: 2
ID: 2110141; N: 1	ID: 1101131; N: 4	ID: 1121151; N: 1	ID: 1002131; N: 1	ID: 0022132; N: 1
ID: 0110121; N: 3	ID: 2101141; N: 1	ID: 1121142; N: 4	ID: 1002122; N: 2	ID: 1022142; N: 1
ID: 1110122; N: 6	ID: 0101121; N: 3	ID: 2121152; N: 1	ID: 2002132; N: 1	ID: 1022133; N: 2
ID: 2110132; N: 3	ID: 1101122; N: 6	ID: 0121132; N: 3	ID: 0102131; N: 1	ID: 2022143; N: 1
ID: 0210131; N: 1	ID: 2101132; N: 3	ID: 1121133; N: 6	ID: 1102141; N: 1	ID: 0122142; N: 1
ID: 1210141; N: 1	ID: 0201131; N: 1	ID: 2121143; N: 3	ID: 1102132; N: 3	ID: 1122152; N: 1
ID: 1210132; N: 3	ID: 1201141; N: 1	ID: 0221142; N: 1	ID: 2102142; N: 1	ID: 1122143; N: 3
ID: 2210142; N: 1	ID: 1201132; N: 3	ID: 1221152; N: 1	ID: 0102122; N: 2	ID: 2122153; N: 1
ID: 0010111; N: 2	ID: 2201142; N: 1	ID: 1221143; N: 3	ID: 1102123; N: 3	ID: 0122133; N: 2
ID: 1010112; N: 3	ID: 0011130; N: 1	ID: 2221153; N: 1	ID: 2102133; N: 2	ID: 1122134; N: 3
ID: 2010122; N: 2	ID: 1011140; N: 1	ID: 0001111; N: 2	ID: 0202132; N: 1	ID: 2122144; N: 2
ID: 0110112; N: 3	ID: 1011131; N: 4	ID: 1001112; N: 3	ID: 1202142; N: 1	ID: 0222143; N: 1
ID: 1110113; N: 4	ID: 2011141; N: 1	ID: 2001122; N: 2	ID: 1202133; N: 2	ID: 1222153; N: 1
ID: 2110123; N: 3	ID: 0111140; N: 1	ID: 0101112; N: 3	ID: 2202143; N: 1	ID: 1222144; N: 2
ID: 0210122; N: 2	ID: 1111150; N: 1	ID: 1101113; N: 4	ID: 0012131; N: 1	ID: 2222154; N: 1
ID: 1210123; N: 3	ID: 1111141; N: 5	ID: 2101123; N: 3	ID: 1012141; N: 1	ID: 0000101; N: 1
ID: 2210133; N: 2	ID: 2111151; N: 1	ID: 0201122; N: 2	ID: 1012132; N: 3	ID: 1000102; N: 1