rijksuniversiteit groningen

faculteit Wiskunde en Natuurwetenschappen

Finding new digits to replace the letters A-F in the hexadecimal numeral system

Bacheloronderzoek Wiskunde

Juli 2013

Student: Anass Bouisaghouane

Eerste Begeleider: P.R.O.F. A.V.Kiselev

Tweede Begeleider: P.R.O.F. H.Waalkens

Abstract

The numbers and letters 0-9 and A-F are each given a graph representation using a 'double box with two diagonals' template, consisting of vertices and edges. All possible digits that fit inside this template are then created, after which some are chosen according to some axioms. A digit is an ordered set of vertices and edges within this template. We define a metric, and try to replace the digits A-F with new digits that satisfy certain conditions. In the end, we discuss some solutions to this problem, and their behaviour.

Contents

1 The origin of the digits

1.1 The need to count

Most ancient civilizations used number systems which described only a small amount of numbers. They did not count beyond certain large numbers, instead they would substitute a large number with the word "many". A modern example in the 19th century was the tribesman of the African Dammara Tribe, who knew that the value of two sticks of tobacco was equivalent to the value of one sheep. The tribesman was however not able to understand that four sticks were equivalent to two sheep. For counting with large numbers, one needs a base system consisting of symbols, each representing an amount. For recording these symbols one of course needs some kind of an alphabet to write them down, or knots tied on strings, as was done by the Chinese and the Incas. [4]

1.2 Sumerian and Babylonian numerals

Some of the first numerals came from the Sumerians in ancient times. Later on they started using the cuneiform (wedge shaped) symbols, which where also adopted by the Babylonians. Some old civilizations, such as the Babylonians, used to associate a number with each one of their deities, in their case 11 in total.[5] The higher the number, the more important the deity was. The Sumerians were some of the first to use a positioning system. [1]
By using a positioning system, large numbers could be represented efficiently using the base of 60 digits. A system with base 60 is called a sexagesimal system. The first position for a number represents units from 0 up to 59, the second for numbers from 60 to $60^2 - 1$, and so on. For numbers up to 200, the values of the symbols are added. In the table below, one can see that the number 35 is constructed by using three symbols for 10 and adding to this five symbols for 1. For larger numbers, the values of the symbols had to be multiplied. For example, 1000 is equal to the symbol for the number 10 next to the symbol for the number 100. The Babylonians also used a symbol or a blank space for zero, although it was not used for calculation. [1] Note that although a positioning system is used, some numerals are difficult to write down, like the number 35.

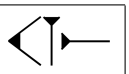
					
1	5	10	100	35	1000

Table 1: Some numbers in cuneiform script.

1.3 Hindu-Arabic numerals

The numerals we are familiar with are known as the Hindu-Arabic numerals. These numerals are thought to have originated from the early Hindus, where they were also known as the Brahmi numerals. These numerals would have been transmitted to the Arab world, after which they were further developed by mathematicians and scientists. By means of teaching calculations, these numerals spread around all the way to the west of the Arab world, where they were probably first introduced in Spain to Europeans that went to study there. Another great contribution to the spreading of the numerals was by Leonardo of Pisa. As a boy he had learned the numerals in North-Africa, and used them in arithmetic and algebra in his book "Liber abaci". [1]

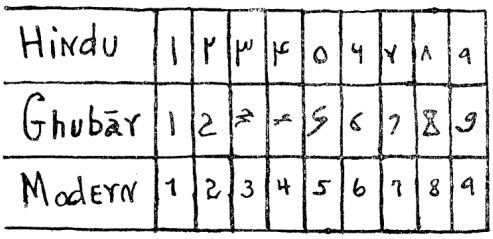


Figure 1: From top to bottom: Hindu-Brahmi numerals, old Arab-Ghubar numerals and modern numerals. [7]

1.4 Controversy

There are many opinions on where the Hindu-Arabic numerals originate from. Some scholars believe that they were developed from the Brahmi numerals.[8] Others suggest that the numerals are developed from the Arabic alphabet. One reason for this could be the fact that many civilisations used to associate letters from their alphabet with numbers. Examples of this are the Greek, the Roman, the Hebrew and the early Arab numerals. The Hindus however, had special symbols for their numerals, not corresponding to the alphabet.[6] [2]
It is also stated in [3], that the numerals we now refer to as the Hindu-Arabic numerals, could have some other origin. They might have originated from the Greek world. The problem with originating the numerals, is that the decimal system, the zero numeral and the positioning system were used widely in ancient times.
Other arguments supporting some other origin of the numerals are given by C.Boyer in [2]. One argument is the translation of the word "hindasi" to "hindi", or "belonging to the hindus", whilst the word hindasi could have many translations in different contexts. An other argument is the fact that some ancient inscriptions on which the original theories were based, have been found to be unauthentic reproductions.

2 Introduction to the problem

In this section, some definitions are given and the problem is described.

2.1 Graphs and Vectors

There are 16 base digits in the hexadecimal system, namely: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, and F. Each of these digits can be given a graph representation, by associating with each digit a graph that lies inside of the graph template. This graph template is a double-box grid with two diagonal lines, displayed in the figure below.

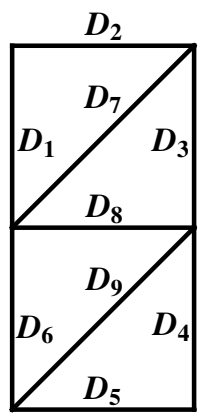


Figure 2: The graph template.

Each of the numbered sides on the graph-template is an edge. The end parts of an edge are called vertices. Note that there are nine edges in the graph-template and six vertices. A digit consist of a set of edges with the associated vertices. Since a digit is completely determined by the nine edges, it is possible to associate with each digit a vector of nine elements. We first number the edges in the graph-template. One way to number the edges, is shown in the above figure. We associate with each digit a vector in the following way:

$$\begin{pmatrix} D_1 \\ D_2 \\ D_3 \\ D_4 \\ D_5 \\ D_6 \\ D_7 \\ D_8 \\ D_9 \end{pmatrix}$$

One can associate a vector with each digit by defining the elements of the vector, namely D_i, to be equal to either zero or one. The element D_i equals 0 if a vertex is not contained in the graph representation of the digit and it equals 1 if the vertex is contained in the graph representation of the digit.

$$D_i = \begin{cases} 1 & \text{if the } i^{th} \text{ vertex is contained in the digit} \\ 0 & \text{if the } i^{th} \text{ vertex is not contained in the digit} \end{cases}$$

Since there are nine vertices in the graph-template and each of them can be contained in the graph representation of a digit, corresponding to a 1 in the associated vector, or not contained in the graph representation of a digit, corresponding to a zero in the associated vector, there is a total number of $2^9 = 512$ digits that fit inside of the template. This set does include the empty graph, i.e. the graph that does not contain any edges.

2.2 The problem

The graph-representations of the digits zero to nine and the letters A to F are chosen to be as described in figure 3 below. They are also referred to as the post office digits. Such digits were used in the past to specify postal codes on letters.

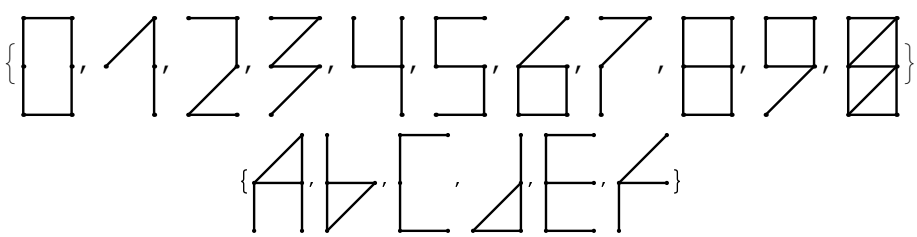


Figure 3: The post office digits as used in the past on letters.

We now pick the graph representations for the numbers 0 up to 9 and the graph-template and fix these. The objective is to try and find six new digits from the remaining set of $512 - 11 = 501$ digits. These six new digits should differ as much as possible from the fixed eleven digits. They should also differ as much as possible from each other, i.e. the digits need to be somehow 'easily recognizable'. The solution of six digits should also be a solution that is better than the solution of the letters A up to F. We will now define what the phrases 'differ as much as possible' and 'a better solution' mean.

3 The Problem

In this section some axioms will be introduced on digits that lie on the graph template, solutions will be discussed and new assumptions are made. Finally the problem formulation will be completed.

3.1 Metric

We start by defining a distance function on the vector of nine digits, representing a digit in graph representation. We first define a function. Suppose we have two vectors $D1$ and $D2$, each containing nine elements, each element being equal to either 0 or 1. The following function associates with each two such vectors and a number $\lambda \in \mathbb{R}_{\geq 1}$, a real number, which we will refer to as the d_λ-distance between D1 and D2.

$$d_\lambda(D1, D2) = \left(\sum_{i=1}^{9} |D1_i - D2_i|^\lambda \right)^{\frac{1}{\lambda}} \tag{1}$$

The number λ is a fixed number that is greater than or equal to one, that can be chosen. The elements $D1_i$ and $D2_i$ equal either 0 or 1. Therefore their difference equals 0 if $D1_i = D2_i$ and their difference equals 1 if $D1_i \neq D2_i$. Taking an absolute value, means that $|D1_i - D2_i|$ is equal to either 0 or 1. Therefore, raising this element to the power of λ still yields a 0 or a 1. This action of raising an element to a power can therefore be omitted during computations from now on. The above equation now simplifies to:

$$d_\lambda(D1, D2) = \left(\sum_{i=1}^{9} |D1_i - D2_i| \right)^{\frac{1}{\lambda}} \tag{2}$$

Since we want a function that expresses the distance between two vectors in some real number, this function must satisfy the axioms of a distance function, or metric. A function that associates with each two elements a number, is called a metric if it satisfies the following axioms. [10]

Definition 1. *Suppose we have a non-empty set X. A function $f : X \times X \to \mathbb{R}$ is a metric, if it satisfies the following 4 axioms:*

1. $f(X, Y) \geq 0$ *(non-negativity)*
2. $f(X, Y) = 0 \iff X = Y$ *(coincidence)*
3. $f(X, Y) = f(Y, X)$ *(symmetry)*
4. $f(X, Z) \leq f(X, Y) + f(Y, Z)$ *(triangle inequality)*

In the above definition, X is the set of all vector representations of a digit. The set $X \times X$ contains element of the form (a,b), where a and b are both elements of the set X. Applying the function f to the set $X \times X$, corresponds to applying

the function f to all elements of the form (a,b). Doing this, one obtains a number that is called the distance between the elements a and b.

3.2 The function d_λ

We will now show that the function d_λ given above satisfies the axioms for a metric. We begin with the first statement. The element $|D1_i - D2_i|$ is always non-negative. The sum of such elements is still non-negative. Taking the λth root of this sum, yields a non-negative number. Therefore the function d_λ satisfies the first axiom.
The second statement is true as well, since the distance between two elements is zero, if and only if the difference between each of their components, being equal to either zero or one, are equal:

$$d_\lambda(D1, D2) = 0 \iff |D1_i - D2_i| = 0 \ \ \forall i$$

Since $|D1_i - D2_i|$ is non-negative, the last statement holds if and only if the elements $D1_i$ and $D2_i$ are equal to each other:

$$D1_i = D2_i \ \ \forall i$$

Therefore, the distance between two digits is zero, if and only if the two digits are equal to each other.
For proving the third statement, we use the fact that the absolute value function is symmetric:

$$|D1_i - D2_i| = |D2_i - D1_i| \tag{3}$$

Taking the sum of such elements, does not change the equality:

$$\sum_{i=1}^{9} |D1_i - D2_i| = \sum_{i=1}^{9} |D2_i - D1_i| \tag{4}$$

Taking the λth root on both sides, also does not change the equality. We therefore have that $d_\lambda(D1, D2) = d_\lambda(D2, D1)$.
The fourth inequality is more difficult to prove. We start with a definition.

Definition 2. *A function $h : [0, \infty] \to [0, \infty]$ is said to be concave on the interval I, if for every line connecting two points on the graph of h, this line lies below the graph of h.*

We will also need the following theorem:

Theorem 1. *A function $h : [0, \infty] \to [0, \infty]$ is said to be concave if the second derivative of h is non-positive: $\frac{d^2h}{dx^2} \leq 0$*

Proof. See [13]. □

We choose the function $g(x) = x^{\frac{1}{l}}$ for non-negative x, where the parameter l is a real number greater than or equal to one. The first derivative of g with respect to the variable x is:

$$\frac{dg}{dx} = \frac{1}{l}x^{\frac{1}{l}-1} = \frac{1}{l}x^{\frac{1-l}{l}} \tag{5}$$

Since $l \geq 1$, $0 < {}^1/_l \leq 1$. This means that $\frac{dg}{dx} > 0$, $g(x)$ is a monotone increasing function. If $a \leq b$ for some real numbers a and b, then $g(a) \leq g(b)$. The second derivative is:

$$\frac{d^2g}{dx^2} = \frac{1}{l}\frac{1-l}{l}x^{\frac{1-l}{l}-1} = \frac{1}{l}\frac{1-l}{l}x^{\frac{1-2l}{l}} \tag{6}$$

Since l is greater than or equal to 1 and x is non-negative, we have the following inequalities on $I = [0, \infty)$:

$$\frac{1}{l} \geq 1 \tag{7}$$

$$\frac{1-l}{l} \leq 0 \tag{8}$$

$$x^{\frac{1-2l}{l}} \geq 0 \tag{9}$$

$$\tag{10}$$

The product of these inequalities yields: $\frac{d^2g}{dx^2} \leq 0$. Therefore the function g is concave on $I = [0, \infty)$. We now look at a consequence of the fact that the function g is concave.

Theorem 2. *If a function $h : [0, \infty] \to [0, \infty]$ is concave on the interval I, then it satisfies the following inequality: $g((1-\kappa)a + \kappa b) \geq (1-\kappa)g(a) + \kappa g(b)$ $\forall \kappa \in [0, 1]$ and $(a, b) \in I$.*

Proof. See Appendix B. □

The function g satisfies the equality $g(0) = 0$. Substituting the value $a = 0$ in the above theorem, yields:

$$g(\kappa b) = g((1-\kappa)0 + \kappa b) \geq (1-\kappa)g(0) + \kappa g(b) = \kappa g(b) \tag{11}$$

From this, it follows that:

$$g(\frac{x+y}{x+y}x) + g(\frac{x+y}{x+y}y) \geq \frac{x}{x+y}g(x+y) + \frac{y}{x+y}g(x+y) \tag{12}$$

Collecting terms with the same denominator yields on the right hand side:

$$\frac{x+y}{x+y}g(x+y) = g(x+y) \tag{13}$$

Since $g(\frac{x+y}{x+y}x) + g(\frac{x+y}{x+y}y) = g(x) + g(y)$, we now have the following inequality:

$$g(x) + g(y) \geq g(x+y) \tag{14}$$

This inequality is also known as the sub-additivity property.
Combining all the the above, one can prove the triangle inequality for the function d_λ. We start by taking three vectors: a, b and c. We want to show that the distance from a to c is less than or equal to the distance from a to b plus the distance from b to c. We use the triangle inequality of the absolute value function to obtain the first inequality.

$$\sum_{i=1}^{9} |a_i - c_i| \leq \sum_{i=1}^{9} |a_i - b_i| + \sum_{i=1}^{9} |b_i - c_i| \tag{15}$$

We write down the distance between a and c according to the function d_λ explicitly and use the fact that the function g is monotone increasing:

$$d_\lambda(a,c) = \left(\sum_{i=1}^{9} |a_i - c_i|)\right)^{\frac{1}{\lambda}} \leq \left(\sum_{i=1}^{9} |a_i - b_i| + \sum_{i=1}^{9} |b_i - c_i|)\right)^{\frac{1}{\lambda}} \tag{16}$$

Applying the sub-additivity inequality to the last expression yields:

$$\left(\sum_{i=1}^{9} |a_i - b_i| + \sum_{i=1}^{9} |b_i - c_i|)\right)^{\frac{1}{\lambda}} \leq \left(\sum_{i=1}^{9} |a_i - b_i|)\right)^{\frac{1}{\lambda}} + \left(\sum_{i=1}^{9} |b_i - c_i|)\right)^{\frac{1}{\lambda}} \tag{17}$$

Where the last term equals $f_\lambda(a,b)+f_\lambda(b,c)$. This yields the triangle inequality: $f_\lambda(a,c) \leq f_\lambda(a,b) + f_\lambda(b,c)$. [12] [11]

3.3 Axioms for the digits

Before adding digits from the set of 501 digits to the eleven fixed digits, we do not want certain digits to be contained in our solutions. To exclude some of the 'bad' digits, we introduce some axioms.

Axioms. *A digit is admissible if it satisfies the following three axioms:*

A1 *The digit is path-connected*

A2 *The digit has width equal to 1*

A3 *The digit has height equal to 2*

The path-connectedness is similar to the connectivity condition for a graph, i.e. suppose we have a graph D with a set of edges E and a set of vertices V. The graph D is connected if, starting from any of the vertices contained in V, there is a path, i.e. a finite collection of edges in E, such that any other vertex in V can be reached. In practice this means that any of the connected digits can be drawn using a pencil and paper, without removing the pencil from the paper. Whilst drawing, the pencil and the paper are always in contact.
Having width 1, means that there has to be at least one horizontal or diagonal edge present in the graph representation of the digit. This means that at least

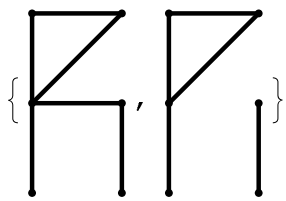


Figure 4: On the left a connected and on the right a disconnected digit.

one of the following five vector components must be non-zero: D_2, D_5, D_7, D_8, or D_9.
Having height 2, means that there is at least one combination of two vector components, each contained in one of the following two sets, that is non-zero: D_1, D_3, D_7 and D_4, D_6, D_9.
The second and third axioms ensure that there are no admissible digits, that do not occupy the complete range of the height and width of the graph template. This ensures that all of the admissible digits have the same "appearance". This might make the digits easier to recognize for systems like automatic handwriting recognition.

3.4 Admissible digits

Using the above axioms and implementing them into a computer program, using Mathematica for example, one can compute the total number of digits that satisfy the axioms. An other way would be to write down manually all these possibilities, by starting with solutions with two vertices, three vertices, up to solutions with nine vertices. Both methods will yield a total number of 333 solutions that satisfy the axioms. One must add to the digits 0 up to 9 and the graph template six new digits to obtain a solution. This way, there is a total of $\binom{333-11}{6} = 1477254941008$ possible solutions. Trying to find the "best" solution or solutions from this many possibilities, would be a Herculean task. The total number of candidate-solutions is simply unmanageable. We are however not interested in all possible solutions. One could state that the only solutions we are interested in, are those that have sufficiently large distances in between themselves and sufficiently large distances in between the old digits and the new digits.

3.5 Further constraints

3.5.1 The original solution A-F

One can take each of the digits in the set that contains the digits 0 to 9, the graph-template and the digits A-F, and compute the distance between this digit and the remaining 16 digits. These distances can then be placed in a matrix. In this case, this is done by making a 17-by-17 matrix. The element on the ith row

and j^{th} column of this matrix must correspond to the distance between the ith digit in the set of 17 digits and the jth digit in the same set. Since the distance function is symmetric, the matrix is also symmetric, i.e. the element in the ith row and j^{th} column is equal to the element in the jth row and i^{th} column. One can therefore omit the distances on either the positions above or the positions below the diagonal. Also, each element on the ith row and i^{th} column is zero, since the distance between an element and itself is zero. This means that the diagonal of the matrix is zero. The symmetry and the zero diagonal yield a matrix with only digits on the positions above the diagonal.

$$\left(\begin{array}{ccccccccccc|cccccc}
0 & 5 & 4 & 8 & 4 & 3 & 5 & 5 & 1 & 5 & 3 & 5 & 6 & 2 & 4 & 3 & 7 \\
0 & 0 & 5 & 5 & 3 & 6 & 4 & 4 & 6 & 6 & 6 & 2 & 7 & 7 & 3 & 8 & 4 \\
0 & 0 & 0 & 4 & 6 & 5 & 7 & 5 & 5 & 3 & 5 & 7 & 6 & 4 & 2 & 5 & 7 \\
0 & 0 & 0 & 0 & 6 & 5 & 5 & 3 & 7 & 3 & 5 & 5 & 4 & 6 & 6 & 5 & 3 \\
0 & 0 & 0 & 0 & 0 & 3 & 5 & 7 & 3 & 3 & 5 & 3 & 4 & 6 & 4 & 5 & 5 \\
0 & 0 & 0 & 0 & 0 & 0 & 4 & 6 & 2 & 4 & 4 & 6 & 5 & 3 & 5 & 2 & 6 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 4 & 4 & 8 & 4 & 2 & 5 & 5 & 5 & 4 & 2 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 6 & 6 & 6 & 4 & 5 & 3 & 7 & 4 & 2 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 4 & 2 & 4 & 5 & 3 & 5 & 2 & 6 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 4 & 6 & 3 & 5 & 5 & 4 & 6 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 4 & 5 & 5 & 5 & 4 & 6 \\
\hline
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 5 & 7 & 5 & 6 & 2 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 4 & 6 & 3 & 3 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 6 & 1 & 5 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 7 & 7 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 4 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{array}\right)$$

Figure 5: The relative distances for the original solution.

We divide the matrix up in upper-left, upper-right and lower-right parts. The upper-left part corresponds to the distances between the fixed 11 digits. The upper-right part corresponds to the distances between each of the digits in the original solution A-F with respect to the fixed 11 digits. The lower-right part of the matrix corresponds to the distances between each of the digits in the old solution A-F with respect to the rest of the digits in the old solution.

3.5.2 Coat of width N

Because of the large number of solutions and the fact that some of them look like each other, we define a new constraint to the digits. This constraint is called "a coat of variable width". Given the 322 admissible digits, one can look at those digits that have a sufficiently large distance with respect to the old 11 digits. One way to do this, is by requiring that the minimal distance must be greater than some chosen integer. Choosing the minimal distance equal to N+1,

corresponds to applying a coat of width N to the old 11 digits, i.e. a coat of width N removes from the 322 digits all those digits that have minimal distance less than or equal to N. Suppose we take the following digit: The d_1-distances

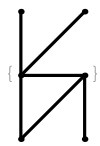


Figure 6: Some digit.

between this digit and the fixed 11 digits are shown in the following vector:

$$\{6, 5, 8, 4, 4, 5, 3, 5, 5, 5, 3\}$$

The minimal distance in this vector is 3. This specific digit therefore satisfies a coat of width 1 and 2. It does not satisfy a coat of width 3 or greater. Applying a coat of width $N = 0, 1, 2, 3$ and $N \geq 4$ to the set of 322 solutions, yields the following numbers of admissible digits:

Coat of width N	Number of remaining digits
$N = 0$	322
$N = 1$	247
$N = 2$	83
$N = 3$	7
$N \geq 4$	0

Note that these sets of digits that satisfy coats of variable width, form nested sets. Indeed, if some digit satisfies a coat of width 3, then the minimal distance for this digit with respect to the old 11 digits is greater than or equal to 4. Therefore the minimal distance is also greater than 3, 2 and 1, and this element would also satisfy the coats of width 1, 2 or 3. The fact that the digits with coat of width 3 yields a set of only 7 candidates, makes it easy to check for solutions in this set. We will discuss this relatively small set later on in the text.
The coat of width N corresponds to a constraint to the upper-right part of the matrix we have seen above. A coat of width N means that the minimal value in this upper-right part of the matrix must be N+1 or greater. Since the minimal distance in this upper-right part of the matrix is 2, we want our new solutions to satisfy at least a coat of width 2.

3.5.3 Minimal relative distance

One other constriction would be to demand a minimal relative distance for the six new digits. This means that each one of the six digits making up the solution, must have a sufficient large distance to each of the five remaining digits. Not only would this decrease the amount of solutions, it would also lead to a greater

difference in the graph representation of the digits. This means that the digits should be easier to distinguish. This constraint corresponds to a constraint on the lower-right part of the 17-by-17 matrix above. Since the minimal distance in the lower-right part of the matrix is 1, we could state that a solution that is better than the original solution must have a minimal value in this part of the matrix greater than or equal to 2.
The value 1 in the lower-right part of the matrix corresponds to the distance between the digits C and E. These digits differ in only 1 edge. Therefore their d_1-distance is equal to 1. We could however change the graph representation for C, in order to change this minimal value. One possibility is shown in the figure below.

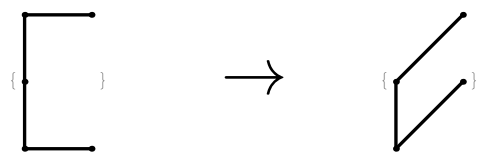


Figure 7: One can replace the old digit for C with a different digit.

Using this new digit for C, the minimal distance that occurs in the lower-right part of the matrix is now equal to 2. We could now state that we would like to have the minimal distance between two digits, in the solution of six digits, to be greater than or equal to 3 in order to obtain a better solution.

3.5.4 Complexity

One last requirement for a solutions of six digits, would be to demand the solutions to have an increasing, or at least non-decreasing complexity. We define the complexity to be the number of rings in a digit. The digit 8 for example has complexity 2, as the digit contain two rings. The full-graph digits has complexity 4, as it contains 4 rings. Note that if we require the solutions of six digits to have non-decreasing complexity, that this is not a restriction. Any set of 6 digits can be ordered such that is has non-decreasing complexity. Demanding a strictly increasing complexity would be a restriction, since there are only digits with complexity equal to 0, 1, 2, 3, or 4. But we need 6 digits, therefore a strictly increasing complexity would be impossible. One could also state a requirement for a certain number of digits with a certain complexity. For example, we could have stated that we want 2 digits of complexity 1, 2 of complexity 2 and 2 digits of complexity 3. We have not made this requirement of complexity during our computations, but it has been done in an article by R.Grubbström [9].

4 Solutions

In this section, some results and solutions to the problem will be discussed.

4.1 The coat of width 3

We will now take a closer look at the set of digits that satisfy a coat of width 3. It has already been stated that there are only seven digits contained in this set with coat of width 3. These seven digits look as follows:

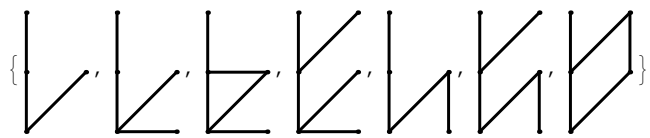


Figure 8: The seven digits for the coat of width 3.

Because there are only 7 digits contained in this set, it is relatively easy to check whether there is a solutions of 6 digits contained in this set. One way of checking for solutions, is by making a matrix of relative distances. The matrix below is the matrix of relative distances for the set of digits with coat of width 3. Note that there are many digits, that have a d_1-distance to another digit equal to 1 in this set of 7 digits.

$$\begin{pmatrix} 0 & 1 & 2 & 2 & 1 & 2 & 2 \\ 0 & 0 & 1 & 1 & 2 & 3 & 3 \\ 0 & 0 & 0 & 2 & 3 & 4 & 4 \\ 0 & 0 & 0 & 0 & 3 & 2 & 2 \\ 0 & 0 & 0 & 0 & 0 & 1 & 3 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \tag{18}$$

One way to check for the solutions of six digits, is by making all $\binom{7}{6} = 7$ subsets of six digits of the set of seven digits. These matrices can be found below.

Note that for every possible subset of six digits of the set of seven digits, there are always two digits with minimal d_1-distance equal to one. Indeed, checking the 7 matrices below, one can see that the element 1 is contained in each of the matrices. Therefore none of the solutions below are admissible, since it has been stated that the minimal distance between two digits in a solutions must be greater than or equal to 2. It is however possible to create a subset of five digits that have relative distances greater than or equal two 2. For example: taking the subset of the first, third, fourth, sixth and seventh digit, we obtain a set of 5 digits, that have minimal relative distances equal to 2.

Out of all possible $\binom{7}{5} = 21$ different subsets of 5 digits, this is the only solution with relative distances ≥ 2.

$$\begin{pmatrix} 0 & 1 & 2 & 2 & 1 & 2 \\ 0 & 0 & 1 & 1 & 2 & 3 \\ 0 & 0 & 0 & 2 & 3 & 4 \\ 0 & 0 & 0 & 0 & 3 & 2 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

(a) Digits $1, 2, 3, 4, 5, 6$.

$$\begin{pmatrix} 0 & 1 & 2 & 2 & 1 & 2 \\ 0 & 0 & 1 & 1 & 2 & 3 \\ 0 & 0 & 0 & 2 & 3 & 4 \\ 0 & 0 & 0 & 0 & 3 & 2 \\ 0 & 0 & 0 & 0 & 0 & 3 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

(b) Digits $1, 2, 3, 4, 5, 7$.

$$\begin{pmatrix} 0 & 1 & 2 & 2 & 2 & 2 \\ 0 & 0 & 1 & 1 & 3 & 3 \\ 0 & 0 & 0 & 2 & 4 & 4 \\ 0 & 0 & 0 & 0 & 2 & 2 \\ 0 & 0 & 0 & 0 & 0 & 2 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

(c) Digits $1, 2, 3, 4, 6, 7$.

$$\begin{pmatrix} 0 & 1 & 2 & 1 & 2 & 2 \\ 0 & 0 & 1 & 2 & 3 & 3 \\ 0 & 0 & 0 & 3 & 4 & 4 \\ 0 & 0 & 0 & 0 & 1 & 3 \\ 0 & 0 & 0 & 0 & 0 & 2 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

(d) Digits $1, 2, 3, 5, 6, 7$.

$$\begin{pmatrix} 0 & 1 & 2 & 1 & 2 & 2 \\ 0 & 0 & 1 & 2 & 3 & 3 \\ 0 & 0 & 0 & 3 & 2 & 2 \\ 0 & 0 & 0 & 0 & 1 & 3 \\ 0 & 0 & 0 & 0 & 0 & 2 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

(e) Digits $1, 2, 4, 5, 6, 7$.

$$\begin{pmatrix} 0 & 2 & 2 & 1 & 2 & 2 \\ 0 & 0 & 2 & 3 & 4 & 4 \\ 0 & 0 & 0 & 3 & 2 & 2 \\ 0 & 0 & 0 & 0 & 1 & 3 \\ 0 & 0 & 0 & 0 & 0 & 2 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

(f) Digits $1, 3, 4, 5, 6, 7$.

$$\begin{pmatrix} 0 & 1 & 1 & 2 & 3 & 3 \\ 0 & 0 & 2 & 3 & 4 & 4 \\ 0 & 0 & 0 & 3 & 2 & 2 \\ 0 & 0 & 0 & 0 & 1 & 3 \\ 0 & 0 & 0 & 0 & 0 & 2 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

(g) Digits $2, 3, 4, 5, 6, 7$.

Figure 9: Matrices of relative distances for all subsets of length six of the set with coat of width 3.

$$\begin{pmatrix} 0 & 2 & 2 & 2 & 2 \\ 0 & 0 & 2 & 4 & 4 \\ 0 & 0 & 0 & 2 & 2 \\ 0 & 0 & 0 & 0 & 2 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

(a) Distance matrix for the digits 1, 3, 4, 6 and 7.

4.2 Solutions for other coats

For the other coats, we would like to proceed in the same way. There is however a problem. In the case of the coat of width 2, there are 83 candidate digits. Out of this set of 83 digits one would like to find solutions consisting of six digits. There are $\binom{83}{6} = 377447148$ sets of six digits for the coat of width 2 and $\binom{247}{6} = 296672379003$ sets of six digits for the coat of width 1. Running

through all these sets of digits, and checking whether their minimal relative distances are greater than or equal to 2 or 3 can be very difficult as it may take a very long time to check all these possibilities. One could note however, that we are not interested in all solutions. The objective was to find a solution. It would however still be interesting to try and find all solutions and analyse their behaviour. We now introduce three algorithms, a good algorithm, a bad algorithm and a naive algorithm.

4.2.1 The good algorithm

The naive way of finding solutions, would be to run through all possibilities and check whether they satisfy the minimal relative distance condition. This method can however be improved. This is done by starting with the set of eleven fixed digits. We take for example the digits that satisfy a coat of width 2. To the set of eleven fixed digits we add one digit at a a time. Adding one of the 83 digits, one has to check for the remaining 82 digits whether they satisfy a large enough distance with respect to the now fixed twelve digits. These digits are then added to the twelve fixed digits and so on. This way, one can prevent the build up of a bad solution, because one of the digits has a to small distance with respect to the other fixed digits. As an example, suppose we have the two digits below.

Figure 11: We want to see if it is possible to start a solution with each of these 2 digits.

The distances between each of these digits and the fixed 11 digits are:

$$\{6, 5, 8, 4, 4, 5, 3, 5, 5, 5, 3\}$$
$$\{2, 5, 4, 6, 2, 1, 5, 7, 1, 3, 3\}$$

The minimal distance for the first digit is 3, and therefore the first digit can be added to the set of 11 digits. This process is then continued, until there are 6 digits. The minimal distance for the second digit is 1. It does not make any sense to continue adding digits to this set, because this second digit is already a bad digit. In this case, the algorithm looks for a different candidate, if there are any.
Using this method, it may still take a lot of time to find all solutions. In practice, we have not been able to find all solutions, or the total number of solutions. We do know that the number of solutions is at least in the order of 1000. Although

it may take a long time to find all solutions using this method. Running the algorithm for several time intervals t and counting the numbers of solutions yields the following table:

Time t in seconds	Number of solutions
30	1666
60	2266
600	4669

Table 2: Running the good algorithm for t seconds yields various amounts of solutions.

It does not take long to find one good solution. And indeed, there are a lot of good solutions.
This same method can be applied to the digits that satisfy a coat of width 1. The problem is that there are 247 digits that satisfy a coat of width 1. It is therefore very likely that there are more solutions in this case, then for the digits that satisfy a coat of width 2. Applying this method to the 7 digits that satisfy a coat of width 3, yields no solutions containing six digits with minimal distance 2. There is however exactly 1 solution of five digits within the set of 7 digits, which is exactly the one we have discussed before. A good solution obtained with this algorithm is shown in the figure below. This figure satisfies a coat of width 2, and a minimal relative distance greater than or equal to 3. The minimal values in the upper- and lower-right part of the distance matrix are therefore equal to 3. Therefore this solution is better than the solution we started out with.

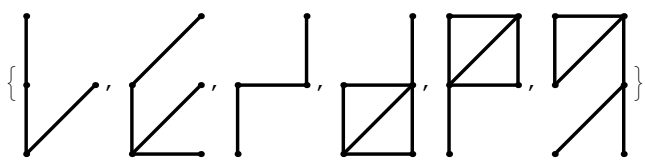


Figure 12: A solution satisfying a coat of width 2 and minimal relative distance greater than or equal to 3.

4.2.2 The bad algorithm

The bad algorithm takes the fixed set of 11 old digits, and computes all the distances of the digits in the set of new digits, for example the coat of width 2, with respect to the old digits. The result is a matrix with dimensions 83 by 11, where each row corresponds to the distance of a new digit with respect to the old 11 digits. The distances can be varied as the metric can also be varied using the parameter λ. The algorithm then takes the sum of all the distances in one row and then finds the maximum sum of distances. Those elements that

satisfy this maximum sum of distances, can then be used to create the solution. Note that there can be multiple digits, corresponding to a row in the matrix, that satisfy this maximum sum of distances. Once a first solution is added, the distances of the remaining 82 digits with respect to the now fixed 12 digits are computed and the process is repeated, until the solution consists of six digits.
Applying this algorithm to the two digits in figure 11, would lead to a sum of d_1-distances of 53 for the first digit and 39 for the second digit. If the maximum of the sum of distances for all candidate digits is equal to 53, then the first digit is added to the set of fixed digits. Note that the second digit is no candidate, because its sum is much smaller.
It would also be possible to require a minimum distance for the digits, as in the good algorithm. This would mean, that in each row of the described matrix, the values need to be greater than or equal to some number. This has not been done in our case. The bad algorithm does decrease the total number of solutions, but this is done in a strange fashion. The behaviour of the number of solutions and the solutions themselves as λ changes, will be discussed later on. This method will yield less solutions, due to the fact that only those digits that lie at a maximal sum of distances from the old digits are added to the list. Although the number of solutions is more manageable, the solutions aren't as good as the ones obtained from the first algorithm, since there is no constraint on the minimal distances. One should also note that the choice has been made to take the sum of the distances, and try to maximize this. This sum is purely artificial. It could also have been possible to take some kind of weighted sum of distances.

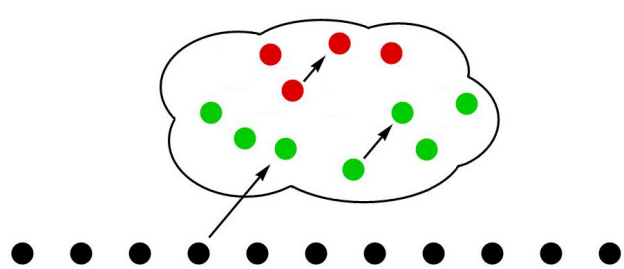


Figure 13: The red dots in the cloud correspond to the digits with coat of width 3. These digits lie further away from the fixed digits, than the digits with a coat of width 2 corresponding to the green dots. But the green dots lie further away from each other than the red dots. The black dots resemble the 11 fixed digits.

An explanation for the bad solutions, given by this algorithm, could lie in the fact that we choose digits with some maximal distance with respect to the fixed digits. These digits have a relatively large distance with respect to the fixed digits, and so their sum is very large. The distances in between these new digits, however, can be very small. This can be seen in the figure above, where the black dots represent the fixed digits, the cloud represents all candidate digits

and the red dots represent the digits chosen in bad solutions. These digits lie close to each other, but very far from the original digits. The green digits, have a smaller distance with respect to the fixed 11 digits, but they lie further from each other than the red digits do. Some solutions obtained with this algorithm are shown in the section below.

4.2.3 The naive algorithm

One final method to find solutions, is in essence much like the good algorithm. We start by taking the fixed 11 digits, and, as an example, the set of 83 digits that satisfy a coat of width 2. We begin by making all 6-tuples of numbers from 1 to 83, such that all numbers differ. That is, we construct elements such as {i,j,k,l,m,n}, where the indices i, j, k, l, m, and n run from 1 to 83. The indices should always differ from each other, in order to ensure that no 6-tuple occurs twice. This can be done by ordering the indices i to n: $i < j < k < l < m < n$. An example of such an element would be {2,4,7,29,64,78}. We associate with this 6-tuple the set of six digits consisting of the ith, the jth, the kth, the lth, the mth and the nth digits. The 6-tuple {2,4,7,29,64,78} corresponds to the digits given in the figure below. The next step is to check whether the 6-tuple

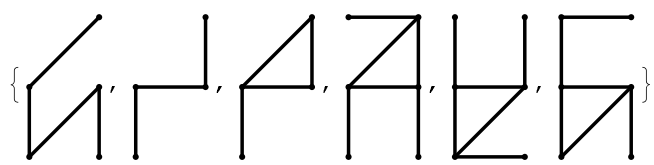


Figure 14: The digits corresponding to the 6-tuple {2,4,7,29,64,78}.

is a solution, by checking for the distances between each of the six digits in the 6-tuple and the remaining 5 digits in the 6-tuple. When a 6-tuple satisfies the minimal relative distance requirement, it is called a solution. The 6-tuple {2,4,7,29,64,78} is not a solution because the minimal relative distance of the 6 digits is 1, which is less than our previous requirement of 2.

$$\begin{pmatrix} 0 & 5. & 4. & 4. & 6. & 4. \\ 0 & 0 & 1. & 3. & 3. & 5. \\ 0 & 0 & 0 & 2. & 4. & 6. \\ 0 & 0 & 0 & 0 & 6. & 4. \\ 0 & 0 & 0 & 0 & 0 & 4. \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Figure 15: The relative distances for the 6-tuple {2,4,7,29,64,78}.

This method is called naive, because it runs through all $\binom{83}{6} = 377447148$ possibilities, before checking any distance. The first algorithm is better, because it

stops constructing a solution, when there is a minimal distance that does not satisfy the requirement of being greater than or equal to 2.
Running the naive algorithm in Matlab for approximately 600 seconds, or 10 minutes, yields a total number 1060210 possibilities that have been checked. This would mean that running through all 377447148 possibilities, would take about 59 hours if the computation time would be extrapolated linearly. However, in practice it has turned out that running the algorithm for millions of solutions, the time actually increases. The expected total computation time should therefore be much greater that 59 hours.

5 Thresholds and behaviour near thresholds

In this section, we will discuss the qualitative and quantitative behaviour of the solutions of the good and the bad algorithm near points where solutions change. These points are called thresholds.

5.1 Thresholds for the good algorithm

Applying the good algorithm to the set of 83 digits that satisfy the coat of width 2 and taking the parameter $\lambda = 1$ in the distance function, yields thousands of solutions that have relative distances greater than or equal to 3. Changing the distance function d_λ whilst computing solutions with the good algorithm on the set of 83 digits that satisfy the coat of width 2, yields changes in solutions. If we require the minimal distance between each of the digits contained in the solution and the remaining 5 to be greater than 2, one yields a huge threshold. If λ is less than this threshold, there are no solutions. If λ is equal to or greater than this threshold, there are thousands of solutions. Changing the required minimal distance, yields a change in the threshold. This can be seen in the table below.

Minimal distance M	Numerical value of threshold
$M \geq 1$	7.6708680×10^{13}
$M \geq 2$	1.5849625
$M \geq 3$	0.9999999
$M \geq 4$	0.7924812
$M \geq 5$	0.6826061

Table 3: Some thresholds for varying minimal distances. For λ smaller than the threshold, there are thousands of solutions. For λ greater than the threshold, there are no solutions.

The threshold seems to behave according to the following relation:

$$\text{threshold} = \frac{\log 3}{\log M} \tag{19}$$

Checking some of the thousands of solutions, we note that there is always a minimal distance in the distance-matrix that equals 3 according to the d_1-metric. This distance is less than four, and therefore, there is no solution if we require the minimal relative distance to be four or greater. This very same distance of 3, changes to the value of 4, according to the $d_{0.7924812}$-metric. The reason for this, is the fact that a d_1-distance can be evaluated into a d_λ-distance by raising the d_1-distance to the power of $1/\lambda$. Raising this distance of 3 to the power of $1/0.7924812$ yields: $3^{\frac{1}{0.7924812}} = 3^{1.26186} = 4$, which satisfies the minimal distance of 4. This explanation holds also for the other minimal distances. By manipulating the minimal distances in the matrix, using the function d_λ no longer as a metric, one can make bad solutions turn into good solutions. Note that for $M \geq N > 3$, the value of the threshold is less than 1. Therefore d_λ is not a metric, since it is no longer a concave function. We can no longer use the theorem that a concave function satisfies the triangle inequality, as the function is no longer concave. This means that the function d_λ does not satisfy the fourth axiom of definition 1, which is the triangle inequality. Therefore d_λ is not a metric any longer.

5.2 Thresholds for the bad algorithm

Applying the bad algorithm to the set of 83 digits that satisfy the coat of width 2, and varying λ results in a change of solutions at different values of λ. There is a quantitative and a qualitative change of the solutions. For example, at $\lambda = 1$ there are 515 solutions. For $\lambda > 1$ the number of solutions decreases rapidly. At $\lambda_1 = 1.7$ there is only 1 solution, we call this solution A. At $\lambda = 3.5$ there is also only one solution, but this solution differs from the one at $\lambda = 1.7$. We call the solution associated with $\lambda = 3.5$, B. Near the value of $\lambda = 3.2438$ both solutions exist. The solutions A and B can be seen in figures 16 and 17.

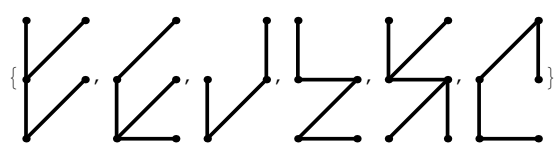


Figure 16: Solution A.

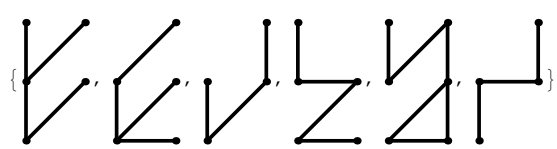


Figure 17: Solution B.

Note that the solutions differ only in the last 2 digits. We now want to know why this change of solutions takes place. For this, we compute the relative distance matrices corresponding to both solutions, by adding each of the solutions A and B to the 11 fixed digits. These distance matrices can be found in Appendix D. Since the solutions start to differ at the sixteenth digit, we start by looking at the sixteenth column, which corresponds to the distances of the sixteenth digit of the solution with respect to the previous 15 digits. These distances are for solution A:

$$\{7, 4, 7, 3, 3, 4, 4, 6, 6, 4, 4, 3, 5, 6, 3, 0, 0\}$$

And for solution B:

$$\{4, 3, 4, 6, 4, 5, 5, 7, 5, 5, 3, 4, 4, 5, 4, 0, 0\}$$

Taking the sum of each of these elements raised to the power of $1/\lambda$ yields two functions, f_A and f_B.

$$f_A(\lambda) = 4 \cdot 3^{\frac{1}{\lambda}} + 5 \cdot 4^{\frac{1}{\lambda}} + 5^{\frac{1}{\lambda}} + 3 \cdot 6^{\frac{1}{\lambda}} + 2 \cdot 7^{\frac{1}{\lambda}}$$

$$f_B(\lambda) = 2 \cdot 3^{\frac{1}{\lambda}} + 6 \cdot 4^{\frac{1}{\lambda}} + 5 \cdot 5^{\frac{1}{\lambda}} + 1 \cdot 6^{\frac{1}{\lambda}} + 1 \cdot 7^{\frac{1}{\lambda}}$$

Evaluating both functions at $\lambda = 1.7, 3.2438$ and 3.5, yields the table above. Near $\lambda = 1.7$, solution A yields the maximum of the sum of $d_{1.7}$-distances. Near

Value of λ	$f_A(\lambda)$	Relative size	$f_B(\lambda)$
$\lambda = 1.7$	36.4017	$>$	36.2748
$\lambda = 3.2438$	23.7767	$=$	23.7767
$\lambda = 3.5$	22.9816	$<$	22.9847

$\lambda = 3.5$, solution B yields the maximum of the sum of $d_{3.5}$-distances. Somewhere in between, there is a threshold. The two solutions occur both at the point where the functions f_A ad f_B coincide. In the figure below, the difference of these two functions is plotted.

$$f_A(\lambda) - f_B(\lambda) = 2 \cdot 3^{\frac{1}{\lambda}} - 4^{\frac{1}{\lambda}} - 4 \cdot 5^{\frac{1}{\lambda}} + 2 \cdot 6^{\frac{1}{\lambda}} + 7^{\frac{1}{\lambda}}$$

Using the method of bisection on the function $f_A(\lambda) - f_B(\lambda)$ yields a zero at $\lambda = 3.2438000281389975028708710438065$. Indeed, at this value, the two functions coincide. Using this approximation of the value of λ at which both functions coincide, yields both solutions A and B. The two solutions both occur, because of round-off errors in the computer. Only at the exact value of the root of the difference of the two functions, both solutions are obtained. This finally explains why there is a value for λ, a threshold, at which both solutions A and B exist. This threshold is the zero of the function $f_A - f_B$.

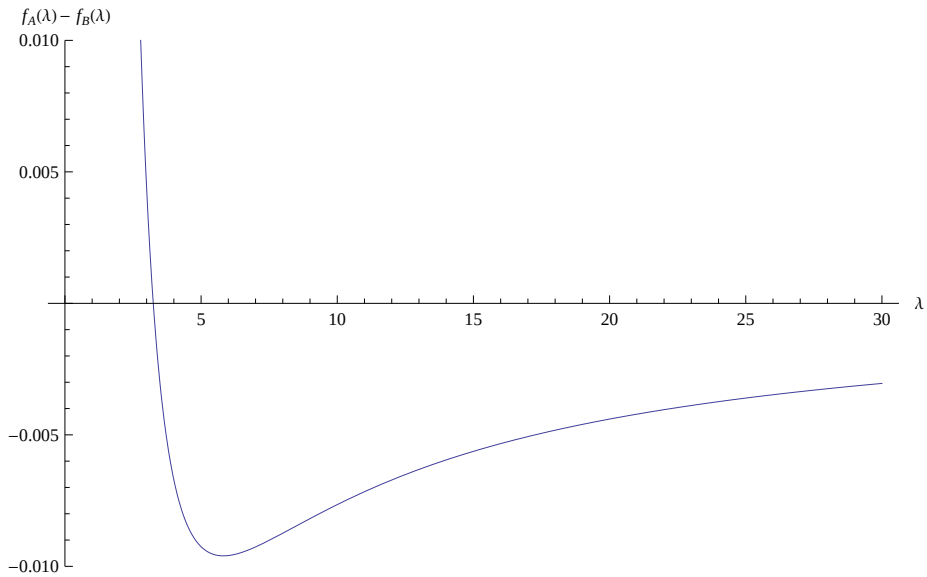


Figure 18: The graph of $f_A - f_B$ for $\lambda \in (0, 30]$.

For the variable λ running from 1 to infinity, the function $f_A - f_B$ crosses the x-axis exactly once. This happens at the threshold we have discussed above. This means that the change of solution from solution A to solution B is unique, at least in the interval $[1.7, 3.5]$ where we have chosen our values for λ. As λ tends to infinity, the function tends towards the x-axis. The x-axis is an asymptote of the function $f_A - f_B$. The function tends towards this asymptote from below.

$$\lim_{\lambda\to\infty}(f_A(\lambda) - f_B(\lambda)) = \lim_{\lambda\to\infty}(2 \cdot 3^{\frac{1}{\lambda}} - 4^{\frac{1}{\lambda}} - 4 \cdot 5^{\frac{1}{\lambda}} + 2 \cdot 6^{\frac{1}{\lambda}} + 7^{\frac{1}{\lambda}}) \tag{20}$$

Since $\lim_{\lambda\to\infty} {}^{1}\!/\lambda = 0$, the above limit simplifies to:

$$\lim_{\lambda\to\infty}(2 \cdot 3^0 - 4^0 - 4 \cdot 5^0 + 2 \cdot 6^0 + 7^0) = 2 - 1 - 4 + 2 + 1 = 0 \tag{21}$$

6 Discussion

We have looked at the origin of the digits, and introduced the concept of digits in a graph template. We have explained why we want to replace the digits A-F, and we introduced a metric, and sever axioms and restrictions to the digits that we wanted to use as a replacement. We have shown three different ways of finding a solution to this problem using algorithms, and we have discussed some of the solutions and their behaviour. In the appendix we also discussed some previous results on a similar problem.
It would be possible to compute all possible solutions, using the naive algorithm, if one has enough time to do this on a computer. This set of all solutions could then perhaps be analysed further, by checking for example if this set is convex. Perhaps it would also be possible to check whether the root of the function $f_A - f_B$, shown in the last section, is rational or irrational.

7 Appendix

7.1 Appendix A: The original problem

In 1992 Prof. Grubbström discussed a problem, similar to ours, in his letter to the editor of The Mathematical Scientist. He started by defining the double box graph.

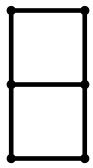


Figure 19: The double box template.

He then gave the digits 0 up to 9 a graph representation, in the following way:

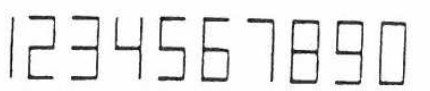


Figure 20: The fixed digits 0-9.

He also gave the digits A to F a graph representation.

Figure 21: The digits A-F

He noted that the digits representing 0 and D are the same. The same holds for the digits 8 and B. He therefore wanted to fix the digits 0-9 and replace the digits A-F with six new digits, such that these digits form a better set than the digits A-F. He did this by defining the following axioms.

Axioms. *A digit is admissible if it satisfies the following five axioms:*

A1 *The digit causes no confusion with existing numerals and roman characters, both upper- and lower-case.*

A2 *The digit is path-connected*

A3 *The digit has height equal to 2*

A4 *The digit must be clearly distinguishable, and as simple as possible*

The first axiom, means that no digits are admissible, if they could be associated with an existing numeral or character in some alphabet. This is to prevent confusion. The second and the third axioms are exactly the same as the ones that have been used in the problem discussed in this paper. Note that the fourth axiom is a subjective axiom. A last requirement is that the 6 digits in the solution must have increasing complexity. Complexity simply means the number of edges contained in the digit. Note that this requirement is not an axiom, since any set of 6 digits can be ordered such that the complexity is non-decreasing.
Since the double box template contains 7 edges, there is a total of $2^7 = 128$ digits that fit inside of the template. Out of this set of 128 digits, the author removes all digits that correspond to existing numerals, characters, those that do not occupy the full height of the template and those that are not path-connected. Out of the set of 128 digits, only 27 remain.

Figure 22: The remaining 27 digits, out of the set of 128 digits, that satisfy the axioms.

The author picks out of this set 6 digits, namely the ones in the figure below, and states that this is a solution. It is not stated how this specific solution is created. Also, the author states that picking a solution would be a subjective matter. Since there are 27 digits, there are now $\binom{27}{6} = 296010$ possible solutions.

Figure 23: A solution for the problem.

This problem originates from the time, when most counters used the double-box template to represent numbers. Nowadays, even the smallest screens contain enough pixels to circumvent the problem of having digits that look like each other. It is however a very nice example of an optimization problem. [9]

7.2 Appendix B: Proof of theorem 2

Suppose that the function h is defined on the interval $[c, d]$ for $c, d \in \mathbb{R}$. We assume that the function h is concave on the interval $I = [a, b]$, where $a, b \in \mathbb{R}$ such that $c \leq a \leq b \leq d$. For the function h to be concave, all the lines connecting any two points on the graph of the function, must lie below the function. In the figure below, one can see the concave function h on the interval [c,d]. A line has been drawn, connecting the two points h(a) and h(b). This line is described by some linear function f. We take a number c $\in \mathbb{R}$ such that $a \leq c \leq b$. Since the function h is concave, the following inequality must hold:

$$f(c) \leq h(c) \quad \forall c \in [a, b] \tag{22}$$

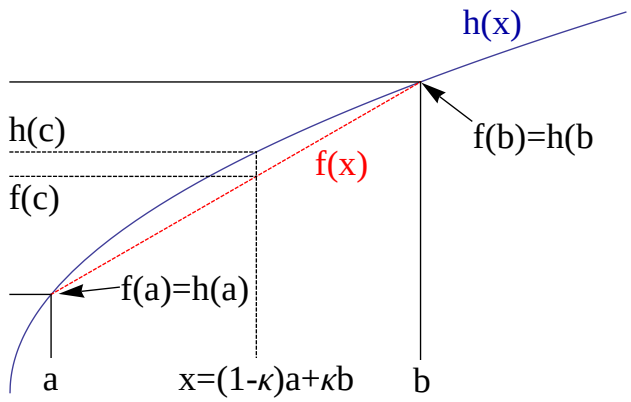


Figure 24: A graph of the functions h and f.

Note that if $a \leq c \leq b$ holds, c can be written as $c = (1 - \kappa)a + \kappa b$ for $0 \leq \kappa \leq 1$. We substitute the rewritten variable c into the function f, and we use the linearity of f:

$$f(c) = f((1 - \kappa)a + \kappa b) = (1 - \kappa)f(a) + \kappa f(b) \tag{23}$$

The functions f and h coincide on the points a and b: $f(a) = h(a)$ and $f(b) = h(b)$. Substituting these equalities into the above equation yields:

$$f(c) = (1 - \kappa)h(a) + \kappa h(b) \tag{24}$$

Since the function h is concave, we have already obtained the inequality in equation 1. Substituting the above equation into equation 1, yields:

$$f(c) = (1 - \kappa)h(a) + \kappa h(b) \leq h(c) \quad \forall c \in [a, b] \tag{25}$$

Using the fact that $h(c) = h((1 - \kappa)a + \kappa b)$, finally yields:

$$h((1 - \kappa)a + \kappa b) \geq (1 - \kappa)h(a) + \kappa h(b) \tag{26}$$

Source: [11] □

7.3 Appendix C: Mathematica error

Whilst trying to find some solutions for the bad algorithm described above, an error was encountered in a Mathematica built in function. This function is called "Position". This function does the following: Suppose we have the vector $V = \{1, 2, 3, 4, 7, 6, 4, 7, 1\}$, and we want to know at which position of this vector a certain value, say 7, is obtained. Plugging this information into the function yields: Position[V,7] = $\{5, 8\}$. This means that the value 7 is obtained at the fifth and eighth position of the vector.
Suppose now that we have the vector V shown below.

$$V = \begin{pmatrix} 18.157 \\ 19.2011 \\ 19.4588 \\ 19.3924 \\ 19.4589 \\ 19.2037 \\ 19.2363 \\ 19.2754 \\ 19.3116 \\ 19.2871 \\ 19.3624 \\ 19.1374 \\ 19.102 \\ 19.1882 \\ 19.4589 \end{pmatrix} \tag{27}$$

We want to find the positions of this vector, at which the maximum of the vector is obtained. This should correspond to the fifth and the fifteenth position: $\{5, 15\}$. Using the Position function however, yields $\{5\}$, $\{15\}$ or $\{5, 15\}$. Running a for-loop over the vector, checking at each position if the maximum is obtained, and adding these positions to an empty set will always yield the output $\{5, 15\}$.

7.4 Appendix D: Figures, Matrices and Mathematica code

The distance matrices used in section 5.2:

$$\left(\begin{array}{ccccccccccc|cccccc}
0 & 5 & 4 & 8 & 4 & 3 & 5 & 5 & 1 & 5 & 3 & 6 & 6 & 5 & 6 & 7 & 4 \\
0 & 0 & 5 & 5 & 3 & 6 & 4 & 4 & 6 & 6 & 6 & 5 & 5 & 4 & 7 & 4 & 3 \\
0 & 0 & 0 & 4 & 6 & 5 & 7 & 5 & 5 & 3 & 5 & 6 & 4 & 3 & 4 & 7 & 4 \\
0 & 0 & 0 & 0 & 6 & 5 & 5 & 3 & 7 & 3 & 5 & 4 & 4 & 5 & 4 & 3 & 6 \\
0 & 0 & 0 & 0 & 0 & 3 & 5 & 7 & 3 & 3 & 5 & 6 & 8 & 5 & 4 & 3 & 6 \\
0 & 0 & 0 & 0 & 0 & 0 & 4 & 6 & 2 & 4 & 4 & 7 & 7 & 8 & 3 & 4 & 7 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 4 & 4 & 8 & 4 & 5 & 3 & 6 & 5 & 4 & 3 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 6 & 6 & 6 & 3 & 3 & 4 & 7 & 6 & 3 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 4 & 2 & 7 & 7 & 6 & 5 & 6 & 5 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 4 & 5 & 7 & 4 & 3 & 4 & 7 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 5 & 5 & 6 & 5 & 4 & 5 \\
\hline
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 3 & 4 & 3 & 4 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 3 & 4 & 5 & 2 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 5 & 6 & 3 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 3 & 6 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 7 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{array}\right)$$

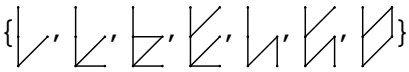
Figure 25: The relative distances for solution A

$$\left(\begin{array}{ccccccccccc|cccccc}
0 & 5 & 4 & 8 & 4 & 3 & 5 & 5 & 1 & 5 & 3 & 6 & 6 & 5 & 6 & 7 & 4 \\
0 & 0 & 5 & 5 & 3 & 6 & 4 & 4 & 6 & 6 & 6 & 5 & 5 & 4 & 7 & 4 & 3 \\
0 & 0 & 0 & 4 & 6 & 5 & 7 & 5 & 5 & 3 & 5 & 6 & 4 & 3 & 4 & 7 & 4 \\
0 & 0 & 0 & 0 & 6 & 5 & 5 & 3 & 7 & 3 & 5 & 4 & 4 & 5 & 4 & 3 & 6 \\
0 & 0 & 0 & 0 & 0 & 3 & 5 & 7 & 3 & 3 & 5 & 6 & 8 & 5 & 4 & 3 & 6 \\
0 & 0 & 0 & 0 & 0 & 0 & 4 & 6 & 2 & 4 & 4 & 7 & 7 & 8 & 3 & 4 & 7 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 4 & 4 & 8 & 4 & 5 & 3 & 6 & 5 & 4 & 3 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 6 & 6 & 6 & 3 & 3 & 4 & 7 & 6 & 3 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 4 & 2 & 7 & 7 & 6 & 5 & 6 & 5 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 4 & 5 & 7 & 4 & 3 & 4 & 7 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 5 & 5 & 6 & 5 & 4 & 5 \\
\hline
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 3 & 4 & 3 & 4 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 3 & 4 & 5 & 2 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 5 & 6 & 3 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 3 & 6 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 7 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{array}\right)$$

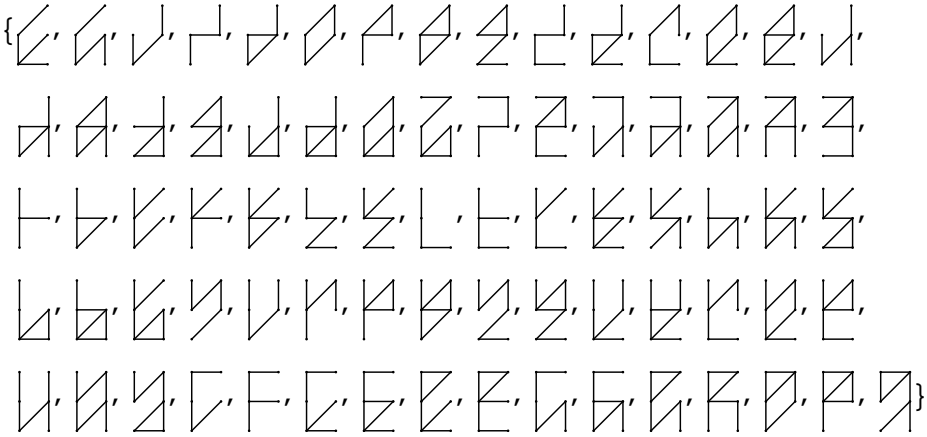
Figure 26: The relative distances for solution B

All digits satisfying coats of variable width, are shown below. The 11 fixed digits are not included, since they have been shown frequently. Below, one can find the 333 axiom satisfying digits and some Mathematica code.

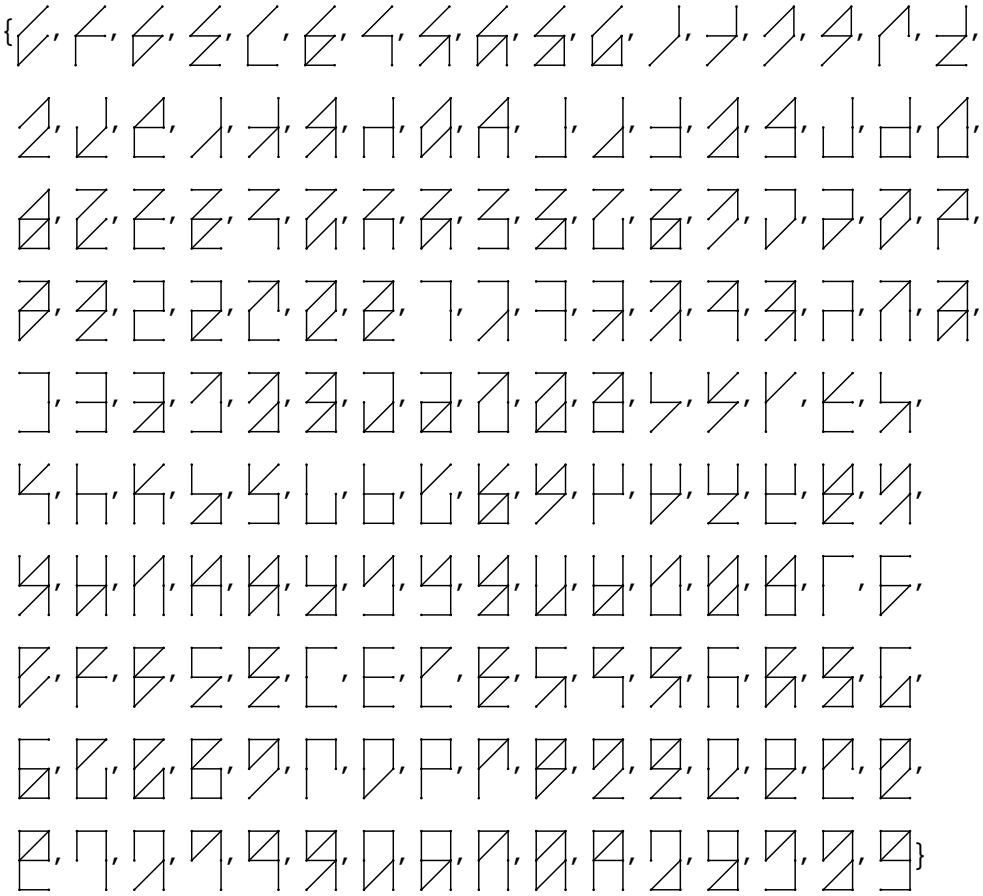
Coat of width 3, without coats of width < 3



Coat of width 2, without coats of width < 2



Coat of width 1, without coats of width < 1



In[6]:=
```
"finding the digits that satisfy the axioms with coat of variable width";
```

In[7]:=
```
SetDirectory[]
G = Import["matrixb.xls"];
G[[1, 512]];
G1 = UnitVector[512, 1];
For[i = 1, i < 513, G1[[i]] = G[[1, i]]; i++];

Clear All;
Clear[l]
Clear[a, c, j, i, n, k, A, B, teller];
teller = 0;
W = UnitVector[333, 1];
For[l = 1, l < 513,
 A = {1, 2, 3, 4, 5, 6};
 q = {1, 2, 3, 4, 5, 6};
 a = G1[[l]];

 If[a[[1]] == a[[6]] == a[[7]] == a[[8]] == 0,
  q[[1]] = Delete[A, 1], q[[1]] = {1, 2, 3, 4, 5, 6}];
 If[a[[1]] == a[[2]] == 0, q[[2]] = Delete[A, 2], q[[2]] = {1, 2, 3, 4, 5, 6}];
 If[a[[2]] == a[[3]] == a[[7]] == 0, q[[3]] = Delete[A, 3], q[[3]] = {1, 2, 3, 4, 5, 6}];
 If[a[[9]] == a[[4]] == a[[3]] == a[[8]] == 0,
  q[[4]] = Delete[A, 4], q[[4]] = {1, 2, 3, 4, 5, 6}];
 If[a[[4]] == a[[5]] == 0, q[[5]] = Delete[A, 5], q[[5]] = {1, 2, 3, 4, 5, 6}];
 If[a[[5]] == a[[6]] == a[[9]] == 0, q[[6]] = Delete[A, 6], q[[6]] = {1, 2, 3, 4, 5, 6}];
 A = Intersection[q[[1]], q[[2]], q[[3]], q[[4]], q[[5]], q[[6]]];


 c = {0, 0, 0, 0, 0, 0, 0, 0, 0};
 j = {0, 0, 0, 0, 0, 0, 0, 0, 0};
 If[a[[1]] == 1, c[[1]] := 1 ⟷ 2]
  If[a[[2]] == 1, c[[2]] := 2 ⟷ 3]
  If[a[[3]] == 1, c[[3]] := 3 ⟷ 4]
  If[a[[4]] == 1, c[[4]] := 4 ⟷ 5]
  If[a[[5]] == 1, c[[5]] := 5 ⟷ 6]
  If[a[[6]] == 1, c[[6]] := 6 ⟷ 1]
  If[a[[7]] == 1, c[[7]] := 1 ⟷ 3]
  If[a[[8]] == 1, c[[8]] := 1 ⟷ 4]
  If[a[[9]] == 1, c[[9]] := 4 ⟷ 6]
  For[i = 1, i < Length[c] + 1, i++, If[c[[i]] == 0, j[[i]] = i]];
 Do[n = 1;
  While[n < Length[j] + 1, If[j[[n]] == 0, j = Delete[j, n]]; n++], {Length[j] + 1}];
 For[k = 1, k < Length[j] + 1, k++, j[[k]] = {j[[k]]}];
 B = Delete[c, j];


 width := (a[[2]] == a[[5]] == a[[7]] == a[[8]] == a[[9]] == 0);
 If[(a[[1]] == 1 && (a[[4]] == 1 || a[[6]] == 1 || a[[9]] == 1)) ||
    (a[[3]] == 1 && (a[[4]] == 1 || a[[6]] == 1 || a[[9]] == 1)) ||
    (a[[7]] == 1 && (a[[4]] == 1 || a[[6]] == 1 || a[[9]] == 1)),
  height = True, height = False];
 connected := (ConnectedGraphQ[Graph[A, B]]);
 If[width == False && height == connected == True && Norm[a] ≠ 0 , W[[teller + 1]] = a]
  If[width == False && height == connected == True && Norm[a] ≠ 0, teller = teller + 1];
 l++]
```

```
teller
Sol = {{1, 1, 1, 1, 1, 1, 0, 0, 0}, {0, 0, 1, 1, 0, 0, 1, 0, 0},
   {0, 1, 1, 0, 1, 0, 0, 0, 1}, {0, 1, 0, 0, 0, 0, 1, 1, 1}, {1, 0, 1, 1, 0, 0, 0, 1, 0},
   {1, 1, 0, 1, 1, 0, 0, 1, 0}, {0, 0, 0, 1, 1, 1, 1, 1, 0}, {0, 1, 0, 0, 0, 1, 1, 0, 0},
   {1, 1, 1, 1, 1, 1, 0, 1, 0}, {1, 1, 1, 0, 0, 0, 0, 1, 1}, {1, 1, 1, 1, 1, 1, 1, 1, 1}};
Normlabda[x_, y_, labda_] := (Sum[(Abs[x[[u]] - y[[u]]])^labda, {u, 1, 9}])^(1 / labda)
Normlabda1[a_, b_] := Sum[Abs[a[[u]] - b[[u]]], {u, 1, 9}]
Normlabda1[G1[[1]], G1[[380]]];

distance = 2
G = W;





Index = 0 * UnitVector[Length[Sol], 1];
For[
 j = 1, j < Length[Sol] + 1,
 counter = 0;
 For[i = 1, i < Length[G] + 1,
  If[Normlabda1[Sol[[j]], G[[i]]] ≤ distance, counter = counter + 1]; i++];
 counter;
 index = 0 * UnitVector[counter, 1];
 counter = 0;
 For[i = 1, i < Length[G] + 1, If[Normlabda1[Sol[[j]], G[[i]]] ≤ distance,
   index[[counter + 1]] = i; counter = counter + 1]; i++];
 index;
 Index[[j]] = index; j++]
Index;


del = Index[[1]];
For[i = 1, i < Length[Index] + 1, del = Union[del, Index[[i]]]; i++]
For[i = 1, i < Length[del] + 1, del[[i]] = {del[[i]]}; i++];


H = Delete[G, del];
Length[G] - Length[H]
If[Length[H] == 0, Print["There are no candidates!"],
 Print[Length[H] "elements satisfy the distance condition"]]
For[j = 1, j < Length[H] + 1,
 For[i = 1, i < 10, If[H[[j]][[i]] == 1, H[[j]][[i]] = 1, H[[j]][[i]] = 0]; i++]; j++]
H;
```

In[36]:= "Graph plotting";

In[37]:=
```
Graafplot[bla_] :=
 (sat = {};
  For[l = 1, l < Length[bla] + 1,
    A = {1, 2, 3, 4, 5, 6};
    q = {1, 2, 3, 4, 5, 6};
    a = bla[[l]];
    bla2 = bla;

    If[a[[1]] == a[[6]] == a[[7]] == a[[8]] == 0,
     q[[1]] = Delete[A, 1], q[[1]] = {1, 2, 3, 4, 5, 6}];
    If[a[[1]] == a[[2]] == 0, q[[2]] = Delete[A, 2], q[[2]] = {1, 2, 3, 4, 5, 6}];
    If[a[[2]] == a[[3]] == a[[7]] == 0,
     q[[3]] = Delete[A, 3], q[[3]] = {1, 2, 3, 4, 5, 6}];
    If[a[[9]] == a[[4]] == a[[3]] == a[[8]] == 0, q[[4]] = Delete[A, 4],
     q[[4]] = {1, 2, 3, 4, 5, 6}];
    If[a[[4]] == a[[5]] == 0, q[[5]] = Delete[A, 5], q[[5]] = {1, 2, 3, 4, 5, 6}];
    If[a[[5]] == a[[6]] == a[[9]] == 0,
     q[[6]] = Delete[A, 6], q[[6]] = {1, 2, 3, 4, 5, 6}];
    A = Intersection[q[[1]], q[[2]], q[[3]], q[[4]], q[[5]], q[[6]]];


    c2 = {0, 0, 0, 0, 0, 0, 0, 0, 0};
    j2 = {0, 0, 0, 0, 0, 0, 0, 0, 0};
    If[a[[1]] == 1, c2[[1]] := 1 -> 2]
     If[a[[2]] == 1, c2[[2]] := 2 -> 3]
     If[a[[3]] == 1, c2[[3]] := 3 -> 4]
     If[a[[4]] == 1, c2[[4]] := 4 -> 5]
     If[a[[5]] == 1, c2[[5]] := 5 -> 6]
     If[a[[6]] == 1, c2[[6]] := 6 -> 1]
     If[a[[7]] == 1, c2[[7]] := 1 -> 3]
     If[a[[8]] == 1, c2[[8]] := 1 -> 4] l
     If[a[[9]] == 1, c2[[9]] := 4 -> 6]
     a;
    c2;
    For[i = 1, i < Length[c2] + 1, i++, If[c2[[i]] == 0, j2[[i]] = i]]
     j;
    Do[n = 1; While[n < Length[j2] + 1,
      If[j2[[n]] == 0, j2 = Delete[j2, n]]; n++], {Length[j2] + 1}];
    j2;
    For[k = 1, k < Length[j2] + 1, k++, j2[[k]] = {j2[[k]]}]
     j2;
    B2 = Delete[c2, j2];
    sat = Append[sat, B2];

    ; l++]

   For[i = 1, i < Length[sat] + 1,
    sat[[i]] = GraphPlot[sat[[i]], VertexCoordinateRules ->
       {1 -> {0, 0}, 2 -> {0, 1}, 3 -> {1, 1}, 4 -> {1, 0}, 5 -> {1, -1}, 6 -> {0, -1}},
      Frame -> False, FrameTicks -> False, VertexLabeling -> False, ImageSize -> 50,
      PlotStyle -> {Black, Thickness -> 0.04, PointSize -> 0.07}] ; i++ ];
  sat)
"The good and the bad algorithm";
```

```
cf1 = Compile[{{x, _Real, 2}},
   motrix2 = 0 * UnitVector[Length[x], 1];
   For[it = 1, it < Length[x] + 1, motrix2[[it]] = Min[x[[it]]]; it++];];
cf = Compile[{{x, _Complex, 1}, {y, _Complex, 1}, {z, _Real}},
   (Sum[Abs[x[[vi]] - y[[vi]]]^(z), {vi, 1, 9}])^(1 / z)];
matrix[Sol2_, ra_] := (
  f[a_, b_] := (cf[H[[a]], Sol2[[b]], ra]);
  Matrix = Array[f, {Length[H], Length[Sol2]}] ;
  cf1[Matrix];
  If[Max[motrix2] > 4,
   Flatten[Position[motrix2, Max[motrix2]]]]
 )
matrix[Sol2_, r_] := (
  f[a_, b_] := (cf[H[[a]], Sol2[[b]], r]);
  Matrix = Array[f, {Length[H], Length[Sol2]}] ;
  Matrix2 = Total[Matrix, {2}] ;
  Flatten[Position[Matrix2, Max[Matrix2]]]
 )
```

```
plops[ra_] := (
  set = {};
  blar1 = matrix[Sol, ra];
  tel = 0;
  Graafplot[Sol];
  For[i1 = 1, i1 < Length[blar1] + 1,
   Sol2 = Sol;
   If[MemberQ[Sol, H[[blar1[[i1]]]]] == False,
    Sol2 = Append[Sol2, H[[blar1[[i1]]]]]];
   blar2 = matrix[Sol2, ra];

   For[j1 = 1, j1 < Length[blar2] + 1,
    Sol3 = Sol2;
    If[MemberQ[Sol2, H[[blar2[[j1]]]]] == False,
     Sol3 = Append[Sol3, H[[blar2[[j1]]]]]];
    blar3 = matrix[Sol3, ra];

    For[k1 = 1, k1 < Length[blar3] + 1,
     Sol4 = Sol3;
     If[MemberQ[Sol3, H[[blar3[[k1]]]]] == False,
      Sol4 = Append[Sol4, H[[blar3[[k1]]]]]];
     blar4 = matrix[Sol4, ra];

     For[l1 = 1, l1 < Length[blar4] + 1,
      Sol5 = Sol4;
      If[MemberQ[Sol4, H[[blar4[[l1]]]]] == False,
       Sol5 = Append[Sol5, H[[blar4[[l1]]]]]];
      blar5 = matrix[Sol5, ra];

      For[m1 = 1, m1 < Length[blar5] + 1,
       Sol6 = Sol5;
       If[MemberQ[Sol5, H[[blar5[[m1]]]]] == False,
        Sol6 = Append[Sol6, H[[blar5[[m1]]]]]];
       "Print[Graafplot[Sol6]]";
       blar6 = matrix[Sol6, ra];

       For[n1 = 1, n1 < Length[blar6] + 1,
        Sol7 = Sol6;
        If[MemberQ[Sol6, H[[blar6[[n1]]]]] == False,
         Sol7 = Append[Sol7, H[[blar6[[n1]]]]]];
        set2 = set;
        For[i2 = 1, i2 < Length[set] + 1, set2[[i2]] = Sort[set[[i2]]]; i2++];
        If[Length[Sol7] == 17 && MemberQ[set2, Sort[Sol7]] == False,
         set = Append[set, Sol7]; tel = tel + 1];

        ; n1++];
       m1++];
      l1++];
     k1++];
    j1++];
   i1++];
  tel)
"Checking for graph-connectedness";
```

In[64]:=
```
connect[c_] := (
  vect = {{1, 2}, {2, 3}, {3, 4}, {4, 5}, {5, 6}, {6, 1}, {1, 3}, {1, 4}, {4, 6}};
  set = {};
  For[s = 1, s < Length[vect] + 1,
   If[c[[s]] == 1, set = Flatten[Append[set, vect[[s]]]]]; s++];
  set1 = {};
  For[t = 1, t < 7,
   For[s = 1, s < Length[set] + 1,
     If[set[[s]] == t, set1 = Append[set1, t]; Break[]]; s++];
   ; t++];
  "Print[set1]";
  set2 = {{}, {}, {}, {}, {}, {}};
  For[i = 1, i < Length[set1] + 1, If[MemberQ[set1, i] == True, set2[[i]] = {i}]; i++];
  "Print[set2]";
 )
```

In[65]:=
```
in[set_, element_] := (
  stat = False;
  For[ri = 1, ri < Length[set] + 1,
    If[element == set[[ri]], stat = True];
    ; ri++]
   stat;
 )
```

In[66]:=
```
functie2[b_, c_, d_] := (
  If[b == 1,
   If[c[[1]] == 1, in[set2[[d]], 2];
    If[stat == False, set2[[d]] = Append[set2[[d]], 2]]];
   If[c[[6]] == 1, in[set2[[d]], 6]; If[stat == False,
     set2[[d]] = Append[set2[[d]], 6]]];
   If[c[[7]] == 1, in[set2[[d]], 3]; If[stat == False,
     set2[[d]] = Append[set2[[d]], 3]]];
   If[c[[8]] == 1, in[set2[[d]], 4]; If[stat == False,
     set2[[d]] = Append[set2[[d]], 4]]];];
  If[b == 2,
   If[c[[1]] == 1, in[set2[[d]], 1];
    If[stat == False, set2[[d]] = Append[set2[[d]], 1]]];
   If[c[[2]] == 1, in[set2[[d]], 3]; If[stat == False,
     set2[[d]] = Append[set2[[d]], 3]]];];
  If[b == 3,
   If[c[[2]] == 1, in[set2[[d]], 2];
    If[stat == False, set2[[d]] = Append[set2[[d]], 2]]];
   If[c[[3]] == 1, in[set2[[d]], 4]; If[stat == False,
     set2[[d]] = Append[set2[[d]], 4]]];
   If[c[[7]] == 1, in[set2[[d]], 1]; If[stat == False,
     set2[[d]] = Append[set2[[d]], 1]]];];
  If[b == 4,
   If[c[[3]] == 1, in[set2[[d]], 3];
    If[stat == False, set2[[d]] = Append[set2[[d]], 3]]];
   If[c[[4]] == 1, in[set2[[d]], 5]; If[stat == False,
     set2[[d]] = Append[set2[[d]], 5]]];
   If[c[[8]] == 1, in[set2[[d]], 1]; If[stat == False,
     set2[[d]] = Append[set2[[d]], 1]]];
   If[c[[9]] == 1, in[set2[[d]], 6]; If[stat == False,
     set2[[d]] = Append[set2[[d]], 6]]];];
  If[b == 5,
   If[c[[4]] == 1, in[set2[[d]], 4];
    If[stat == False, set2[[d]] = Append[set2[[d]], 4]]];
   If[c[[5]] == 1, in[set2[[d]], 6]; If[stat == False,
     set2[[d]] = Append[set2[[d]], 6]]];];
  If[b == 6,
   If[c[[5]] == 1, in[set2[[d]], 5];
    If[stat == False, set2[[d]] = Append[set2[[d]], 5]]];
   If[c[[6]] == 1, in[set2[[d]], 1]; If[stat == False,
     set2[[d]] = Append[set2[[d]], 1]]];
   If[c[[9]] == 1, in[set2[[d]], 4]; If[stat == False,
     set2[[d]] = Append[set2[[d]], 4]]];];
  "Print[set2]";)
```

In[67]:=
```
Sorteer[rij_] := (
  rij2 = {Min[rij]};
  For[tk = 1, tk < 6,
   For[tj = 1, tj < Length[rij] + 1,
     If[rij[[tj]] == rij2[[1]] + tk , rij2 = Append[rij2, rij[[tj]]]]; tj++];
   ; tk++];
  rij2
 )
```

In[68]:=
```
a = {1, 0, 1, 1, 0, 1, 0, 0, 0};
```

In[69]:=
```
Connected[a_] := (
  connect[a];
  For[j = 1, j < Length[set1] + 1,
   If[set2[[j]] ≠ {},
     For[qi = 1, qi < Length[set2[[j]]] + 1, functie2[set2[[j]][[qi]], a, j]; qi++]];
   ; j++];
  set2;
  st = {};
  For[ti = 1, ti < Length[set2] + 1,
   If[set2[[ti]] ≠ {}, st = Append[st, set2[[ti]]]]; ti++];
  st;
  status = True;
  For[j = 1, j < Length[st] + 1, For[k = 1, k < Length[st] + 1, If[k > j, If[
      Sorteer[set2[[k]]] ≠ Sorteer[set2[[j]]], status = False; Break[]]]; k++]; j++];
  status)
```

References

[1] F. Cajori, A HISTORY OF MATHEMATICAL NOTATIONS, The Open Court Company, Vol. I, 1928, http://www.archive.org/details/historyofmathema031756mbp

[2] F. Cajori, The Controversy on the Origin of Our Numerals, American Association for the Advancement of Science, The Scientific Monthly, Vol. 9, no. 5, 1919, 458-464, http://www.jstor.org/stable/6795

[3] C. B. Boyer, Zero: The Symbol, the Concept, the Number, National Mathematics Magazine, Vol. 18, no. 8, 1944, 323-330, http://www.jstor.org/stable/3030083

[4] G. Donald Allen, The Origins of Mathematics, Texas A&M University, Department of Mathematics, 2000, 2003, http://www.math.tamu.edu/ dallen/masters/origins/origins.pdf

[5] Edward Hincks, On the Assyrian Mythology, Royal Irish Academy, Vol. 22, 1849, 405-422, http://www.jstor.org/stable/30079844

[6] Ahmed Boucenna, Origin of the numerals, Al-Birunis testimony, Ferhat Abbas University, Department of Physics, Faculty of Sciences, 2007, http://arxiv.org/ftp/arxiv/papers/0707/0707.3279.pdf

[7] Solomon Gandz, The Origin of the Ghubr Numerals, or the Arabian Abacus and the Articuli, The University of Chicago Press, Vol. 16, 1931, 393-424, http://www.jstor.org/stable/224714

[8] Oystein Ore, Number Theory and its History, McGraww-Hill Book Company , inc., 1948, http://ia600806.us.archive.org/22/items/NumberTheoryItsHistory/Ore-NumberTheoryItsHistory.pdf

[9] Robert W. Grubbström, Thoughts on a new numerical system for hexadecimal numbers., The Mathematical Scientist, Vol. 17, no. 2, 1992, 128-130

[10] Wilson A. Sutherland, Introduction to Metric & Topological Spaces., Oxford University Press, 2nd edition, 2008, 39

[11] Martin J. Osborne, Concave and convex functions of a single variable, University of Toronto, 1997, 2003, http://www.economics.utoronto.ca/osborne/MathTutorial/CV1F.HTM

[12] Concave function, 2010, http://en.wikipedia.org/wiki/Concave_function

[13] Calculus, Early Transcendentals, James Stewart, Brooklyn Cengage Learning, 6th edition, 2008, A44