EÖTVÖS LORÁND UNIVERSITY

DEPARTMENT OF MATHTEMATICS

Zsombor Sárosdi

THE MATHEMATICS OF SUDOKU

Bsc Thesis in Applied Mathematics

Supervisor:

István Ágoston

Department of Algebra and Number Theory

Budapest, 2018

Acknowledgements

I would like to thank my supervisor István Ágoston for recommending this interesting topic and guiding me through it. Apart from the demanding professional assistance I am grateful to him for correcting my English grammar, too. In connection with the style of the language I am also beholden to Emese Sárosdi who helped me out with well-spoken expressions many times.

I am thankful to my friends, especially Renáta Fáy, for reminding me to meet important deadlines and formal requirements.

Contents

Chapter 1

Introduction

The idea of the Sudoku puzzles can be derived from the latin squares. A latin square is an $n \times n$ grid containing the numbers 1 to n in such an arrangement that each number can appear in any row and in any column exactly once. This idea was developed and studied by the famous Swiss mathematician Leonhard Euler (1707-1783). His inspiration came from the magic squares of the ancient China, where we have to write each number from 1 to n^2 into an $n \times n$ square that the sum of the numbers in each row, column and the two diagonals of the square must agree.

Beside the fact that latin squares have many real life applications, French newspapers experimented with creating puzzles by giving some clues from latin squares. The objective was to find the – hopefully unique – completion. This can be clearly considered as the ancestor of the modern Sudoku. Although these puzzles had disappeared after World War I, a few decades later, Howard Garns, a retired architect designed the modern Sudoku. It was published by the Dell Magazine in 1979 under the name Number Place. A few years later the game spread quickly in Japan by the publisher Nikoli, who named the puzzle Sūji wa dokushin ni kagiru, which may be translated as "the digits must be single". It seemed that Sudoku would be confined to Japan until an Australian judge, Wayne Gould saw a puzzle in a bookshop in 1997 and became addicted immediately. He developed and published a computer program which could produce puzzles rapidly and in the very same year, in 2004 The Times started to publish Sudoku puzzles. Due to this achievement Sudoku became wide-spread in the world, and nowadays it is one of the most popular puzzles globally.

The aim of my thesis is to introduce the game from a mathematical point of view.

Chapter 2

General properties

2.1 Number of sudoku grids

In order to avoid misunderstandings let us make clear that under *sudoku grid* or *sudoku table* we mean a 9×9 grid which has a number in each of its cells and it obeys the well-known rules of the Sudoku described below. By the expression *sudoku puzzle* we mean a 9×9 grid which has missing values, but the existing values do not contradict to the rules of the game. In general, we also require that a sudoku puzzle can be completed to a sudoku grid in exactly one way. We may refer to this completion as the *solution* of the sudoku puzzle. According to the rules the sudoku grid is a 9×9 latin square such that every digit from 1 to 9 has to appear in the intersection of the first three rows and first three columns, as well as in the intersection of the first three rows and the second three columns, etc. We will refer to these regions of the sudoku table as *blocks*. Sometimes we will denote them by B1, B2, etc., B9 in such an arrangement that B1, B2 and B3 are the blocks of the first three rows, B4, B5, B6 are the blocks of the second three rows and B7, B8, B9 are the blocks of the last three rows. Sometimes we will refer to a group of the blocks, for example the first *band* of the blocks will refer to B1, B2 and B3, the second band is B4, B5 and B6, and finally the third band is B7, B8 and B9. Similarly, we will refer to the set of first three columns as the first *stack* (B1-4-7), the set of the second three columns as the second stack (B2-5-8) and the remaining three columns as the third stack (B3-6-9).

One of the simplest questions that a mathematician may raise is how many sudoku grids exist. The answer was first published by Bertram Felgenhauer and Frazer Jarvis in 2005 [1]. We will introduce here the method they applied.

Being the sudoku a special case of a latin square – and the enumeration of latin squares is not an easy problem – there is no reason to hope that a closed formula can be obtained, or that any elementary combinatorial reasoning will lead us to the answer. Instead of that, the authors of the article mentioned above developed a computer program to accomplish the harder calculations.

The first observation that they made is that two sudoku grids are essentially the same if one can be obtained from the other by just relabelling the digits (for example exchanging all 1 digits to 4s, all 4 digits to 5s, etc.). Thus, we can count only those tables which have their first block in the canonical form shown by the figure below, and then just multiply the result by 9!, which is the number of the rearrangements of the digits.

1	2	3						
4	5	6						
7	8	9						

2.1.1 The first three blocks

The second step is to fill the following two blocks, B2 and B3. We will refer to a completion of B1-B2-B3 as a *top band completion*. In the remaining six cells of the first row we have to put the numbers from the second two rows of the first block in some order. We will distinguish between two kinds of arrangements: the one where the top row of B2 contains every digit from the second row of B1 and the top row of B3 contains every digit form the third row of B1, or the other way round; and the one where the top row of B2 contains digits from both the second and third row of B1 (and so does B3 as well). We will call these top row arrangements *pure top row* and *mixed top row* arrangements, respectively.

In the case of a pure top row arrangement let us say that we put the numbers of B1′s second row in some order into the top row of B2. Then, we can not choose numbers from the first row of B1 to the second row of B2, because this way we would have to put a third-row digit of B1 (i.e. 7, 8, 9) into the third row of B2, which contradicts to the rules of the game. So the second-row digits of B2 must be the same that the third-row digits of B1. With an analogous argument we can say that B2 and B3 – aside from the order of the numbers in the rows of the latest two blocks – has to look like

1	2	3	4	5	6	7	8	9
4	5	6	7	8	9	1	2	3
7	8	9	1	2	3	4	5	6

Since in each row of B2 and B3 the digits are allowed to be in any order, there are $(3!)^6$ possible arrangements, and considering the other pure top row arrangement, where the top row of B2 contains the numbers 7, 8, 9 we can conclude that there are $2 \cdot (3!)^6$ pure top row arrangements.

Now let us count the number of mixed top row arrangements. We can list all the possible combinations of the numbers in lexicographical order. The first column shows the elements of the top row of B2 and the second contains the top row of B3.

$\{4,5,7\}$	$\{6,8,9\}$
$\{4,5,8\}$	$\{6,7,9\}$
$\{4,5,9\}$	$\{6,7,8\}$
$\{4,6,7\}$	$\{5,8,9\}$
$\{4,6,8\}$	$\{5,7,9\}$
$\{4,6,9\}$	$\{5,7,8\}$
$\{5,6,7\}$	$\{4,8,9\}$
$\{5,6,8\}$	$\{4,7,9\}$
$\{5,6,9\}$	$\{4,7,8\}$

Considering the symmetry of the top rows of B2 and B3 we can conclude that all the other possibilities can be obtained by interchanging these two columns, so there are 18 top row combinations.

Let us see in how many ways can we complete the remaining cells of B2 and B3 from any of these top row situations. Let us first examine the $\{4,5,7\}$ $\{6,8,9\}$ completion of the first row. It is evident that the second row of B2 has to contain the numbers 8 and 9 - in some order - and some element of the first row of B1, let us denote this by f_1. The third row of B2 has to contain the digit 6 and the two other elements of the first row of B1 - let us say f_2 and f_3. We complete the block B3 as follows (all the numbers in the same row of B2 or B3 are considered regardless their order):

1	2	3	4	5	7	6	8	9
4	5	6	8	9	f_1	7	f_2	f_3
7	8	9	6	f_2	f_3	4	5	f_1

Counting all the possible permutations of the rows of B2 and B3 we have $(3!)^6$ possible completions, multiplied by 3 as we can choose f_1, f_2, f_3 in three essentially different ways (the role of f_2 and f_3 is interchangeable). We can do the same argument in any of the 9 combinations of the mixed top rows above, which gives us another 9 multiplier and, as we mentioned, it before the role of B2 and B3 is also interchangeable, so we have to duplicate this result. In conclusion, we have

$$2 \cdot 9 \cdot 3 \cdot (3!)^6$$

mixed top row completions of the first three blocks.

To sum up, having the first block in the canonical form, we have

$$2 \cdot (3!)^6 + 2 \cdot 9 \cdot 3 \cdot (3!)^6 = 56 \cdot (3!)^6 = 2612736$$

ways to complete the first three blocks, and we have $9! \cdot 2612736$ completions of the first band all together.

2.1.2 Reduction

The idea of the program is to loop over these 2612736 cases and determine the number of valid completions to a sudoku grid. However, this would take unreasonably much time, so we need to do some reductions. We will define equivalence classes over the 2612736 completions of the top band blocks so that every element within an equivalence class has the same number of valid completions. There are operations which leave the number of possible completions invariant. The relabeling operation - that we have already applied when we defined the top left block to be in the canonical form - is an example. We will call the next type the *swap* operation, which means swapping of two blocks. Indeed, if we have a completion of B1, B2 and B3, and we swap for example B2 with B3, the number of the valid completions will be the same for the two partial table, since from every completion of the first case we can obtain a completion of the second one by swapping B5 with B6 and B8 with B9. The other operation we will use is the permutation of the columns within B2 or B3. If we do a permutation of the columns of B2 we have to do the same with the columns of B5 and B8 and similarly a permutation of the columns of B3 implies the same permutation on the columns of B6 and B9.

Applying these operations we can obtain a *lexicographically ordered* form of each of the 2612736 grids:

1. First of all we can permute the columns of both B2 and B3 so that within each block the digits are in increasing order.

2. Secondly, we swap B2 and B3 if the first element in the top row of B3 is smaller than the first element of the top row of B2.

Any lexicographically ordered sudoku table is a representative of an equivalence class. The first operation gives $3! \cdot 3! = 36$ possible elements to the equivalence class and the second operation duplicates this number, so we have $2612736/72 = 36288$ equivalence classes.

Let us sum up where we are:

1. We calculated that there are $9! \cdot 2612736$ completions of the first band.

2. We defined the *lexicographically ordered sudoku grid* , which means that B1 is in the canonical form, the elements of the first row of B2 and B3 are in increasing order and the first element of B2 is greater than B3's.

3. Each pair of first band arrangements are equivalent (in the sense that they have the same number of completions) if they have the same lexicographically ordered form.

4. Thus, we have 36288 equivalence classes.

Having these equivalence classes we can observe that there are some groups of equivalence classes among these 36288 which have the same number of completions. Thus, by uniting such classes we can reduce the number of equivalence classes much further. The reason why this is possible is that we did not use all the benefits of the relabelling transformation neither the permutation of the rows. What we can do is to take any permutation of the top three blocks B1, B2 and B3 (this means $3! = 6$ possibilities) and any permutation of the columns within a block (which are $(3!)^3 = 216$ possibilities) and even a permutation of the rows (again, 6 permutations). It is evident that if we do these transformations to the top rows the number of completions remain the same. Having a transformed sudoku grid we can obtain the lexicographically ordered representative by relabelling B1 into the canonical form and applying the transformations that we described before. This way we can come to the observation that two equivalence classes which were defined has the same number of completions, thus can be united.

As an example let us consider the following lexicographical arrangement of the top three blocks:

1	2	3	4	5	8	6	7	9
4	5	6	9	7	2	3	1	8
7	8	9	1	6	3	5	2	4

Let us permute the blocks:

6	7	9	1	2	3	4	5	8
3	1	8	4	5	6	9	7	2
5	2	4	7	8	9	1	6	3

After performing a permutation of the columns in each block we get the following:

7	9	6	3	1	2	8	5	4
1	8	3	6	4	5	2	7	9
2	4	5	9	7	8	3	6	1

As another equivalent transformation we can permute the rows as well:

1	8	3	6	4	5	2	7	9
7	9	6	3	1	2	8	5	4
2	4	5	9	7	8	3	6	1

In order to prove that we arrived to a different equivalence class we now arrange this to its lexicographical form. It is easy to see that the relabelling is represented by the permutation $(59)(8274)$ (so we exchange all 5s to 9s, 9s to 5s and exchange all 8s to 2s, all 2s to 7s, etc), thus we obtain the canonical form of B1:

1	2	3	6	8	9	7	4	5
4	5	6	3	1	7	2	9	8
7	8	9	5	4	2	3	6	1

The numbers in the first row of B2 are already standing in increasing order, but in B3 we have to put the first column to the end:

1	2	3	6	8	9	4	5	7
4	5	6	3	1	7	9	8	2
7	8	9	5	4	2	6	1	3

And finally we need to swap B2 and B3 because the first element of the first row in B3 is smaller than in B2:

1	2	3	4	5	7	6	8	9
4	5	6	9	8	2	3	7	1
7	8	9	6	1	3	5	4	2

Thus we arrived to a lexicographically ordered form which is different from the original one, but since we used equivalent transformations the number of completions of each grid is the same. Thus, here is an example when we can unite two equivalence classes.

So Jarvis and Felgenhauer's computer program does the following. It creates a catalogue of the 36288 lexicographically ordered sudoku grids, then loops over each of them, produces all the possible permutations of B1, B2 and B3 and all the possible permutations of the columns of each box, and all possible row permutations. This is $3! \cdot (3!)^3 \cdot 3! = 6^5 = 7776$ permutations. Then it creates the lexicographically ordered form of the obtained top band completion and it unites the two equivalence classes if they differ (of course, we need to count the elements of the new class). In fact, if we do these operations to a top band completion which has a pure top row arrangement we will not get an element of another equivalence class, so we need to loop over only the mixed top row arrangements.

Jarvis and Felgenhauer stated that using these calculations they reduced the number of equivalence classes from 36288 to only 416, which is a huge improvement.

2.1.3 The result

Using further methods it was possible to decrease the number of equivalence classes to 71, which was small enough to run a program. A representative from each class was chosen – the lexicographically ordered one – and the size of the equivalence class was calculated. Then, the program calculated all possible completions of the representative, this number was multiplied by the size of the equivalence class. After that the sum of these 71 numbers were computed and it had to be multiplied by 9!. It found out that the number of sudoku grids is

$$6670903752021072936960 \approx 6.671 \cdot 10^{21}$$

When the sizes of the equivalence classes were calculated there were 44 different values. That is that there are precisely 44 equivalence classes.

2.2 Minimum number of clues

After counting the number of possible sudoku grids we can move toward the analysis of the sudoku puzzles. We agreed that a sudoku puzzle should have a unique solution, but what conditions can ensure the uniqueness? For example, at least how many clues should a puzzle contain if we want it to have exactly one solution?

The fact that we need more than 7 clues to ensure a unique solution is quite obvious. Indeed, if we have only 7 clues, there are at least two numbers from 1 to 9 that have not emerged, let us say that these are 8 and 9. Assume that we found a solution of the puzzle. If we exchange all the 8s to 9s and vice versa we come to another solution of the same puzzle.

Though no other purely mathematical argument was found for the non-existence of a puzzle with 8 or more clues, sudoku fans have made the conjecture that the minimum required number of the given numbers is 17, since many 17-clue puzzles have been found, but not even one with 16 clues. The figure below shows an example and its solution to a puzzle which contains 17 clues.

			8		1			
							4	3
5								
				7		8		
						1		
	2			3				
6							7	5
		3	4					
			2			6		

2	3	7	8	4	1	5	6	9
1	8	6	7	9	5	2	4	3
5	9	4	3	2	6	7	1	8
3	1	5	6	7	4	8	9	2
4	6	9	5	8	2	1	3	7
7	2	8	1	3	9	4	5	6
6	4	2	9	1	8	3	7	5
8	5	3	4	6	7	9	2	1
9	7	1	2	5	3	6	8	4

This problem remained unsolved until 2013, when a computer-assisted proof was published that there are no sudoku puzzles with unique solution if there are only 16 clues given [4]. In my thesis I do not intend to show how their program works, but there are some ideas which are worth introducing.

The fundamental approach to the problem was based on the following notion of *unavoidable sets* .

Definition 2.1 *For a given sudoku grid, a subset of the cells is called unavoidable set, if deleting its values the remaining table - as a puzzle - has at least two completions. The unavoidable set is said to be minimal if it has no proper subset which is unavoidable.*

The following example illustrates the essence of unavoidable sets. Let us consider the solution of the 17-clue sudoku above:

2	3	7	8	4	1	5	6	9
1	8	6	7	9	5	2	4	3
5	9	4	3	2	6	7	1	8
3	1	5	6	7	4	8	9	2
④	⑥	9	5	8	2	1	3	7
7	2	8	1	3	9	4	5	6
⑥	④	2	9	1	8	3	7	5
8	5	3	4	6	7	9	2	1
9	7	1	2	5	3	6	8	4

The set which is highlighted in red is an unavoidable set, since if we delete its elements and then recomplete the grid we can have two solutions: the original and the one in which the role of the 6 and 4 is swapped only in the red region.

The idea behind the proof of the non-existence of 16-clue puzzles is briefly the following: we need to go through all the possible sudoku grids, find a collection of unavoidable sets, then build all the hitting sets with 16 elements of this collection (a set of the cells which "hit" all the unavoidable sets: each unavoidable set has to contain exactly one element from the hitting set) and test by a solver if the puzzle constructed this way has two or more completion.

It is not hard to see that this concept needs some improvement. For example the authors of the proof achieved that it is sufficient to check only a small collection of unavoidable sets, by which they compensated one of the drawback of their intention. This improvement will not be covered here. I found however mathematically more interesting their method of decreasing the number of sudoku tables which were needed to be checked. Their idea was similar to that in the previous section: they defined equivalence classes on the set of all the sudoku tables so that if a table has a k-clue puzzle then all tables that are equivalent with that also have a k-clue puzzle.

Definition 2.2 *Two sudoku grids are said to be essentially the same if one can be obtained from the other by applying the following transformations:*

1. *Relabelling.*
2. *Taking the transpose of the grid.*
3. *Exchanging two stacks.*
4. *Exchanging two bands.*
5. *Swapping two rows in a band.*
6. *Swapping two columns in a stack.*

It is easy to check that the relation that two grids are essentially the same really is an equivalence relation in the mathematical sense. The following proposition is the heart and soul of the proof.

Theorem 2.3 *If two tables are essentially the same and one has a k-clue puzzle, then the other also has a k-clue puzzle.*

Proof: It is easy to check that if we have a series of transformations taking the first grid to another, then by applying the same transformations to the k-clue puzzle we can obtain a k-clue puzzle in the latter one. □

The reason why this result is important is that it is sufficient to consider only one representative from each equivalence class and to check that if it has a 16-clue puzzle. If the answer is no, then no other member of the class has a 16-clue puzzle.

2.2.1 Number of equivalence classes

The set of the mentioned transformations from definition 2.2 generates a group with respect to the composition. Let us recall the definition of group action.

Definition 2.4 *For a group G and set X we say that* $\cdot : G \times X \to X$ *is a group action of G on X if it satisfies* $1_G \cdot x = x \quad \forall x \in X$ *and* $(gh) \cdot x = g(h \cdot x) \quad \forall g, h \in G, \forall x \in X$.

The group of these transformations clearly acts on the set of sudoku tables and that is why this group is often referred to as the *sudoku group*. Of course, the number of orbits – i.e. the sets $G \cdot x = \{g \cdot x : g \in G\}$ – of this group action is the number of the equivalence classes, so it is an obvious idea to use Burnside's lemma to count them. Let us reccal the definition of the fixed points of X by $g \in G$ and let us denote it by X^g, so $X^g = \{x \in X : g \cdot x = x\}$. Burnside's lemma states the following:

Theorem 2.5 *Let G be a finite group acting on the finite set X. Then the number of orbits can be calculated by the formula*

$$\#orbits = \frac{1}{|G|} \sum_{g \in G} |X^g|$$

Now, our aim is to define the sudoku group and its action on the set of the sudoku tables precisely.

The relabelling transformation can be represented by S_9, and to indicate whether the transpose was taken we can use C_2, the cyclic group with two elements. The group of row operations (swapping bands or swapping rows in a band) is isomorphic to the group of column operations (swapping stacks or swapping columns in a stack). This group can be represented by a subgroup of S_9:

$$T = \left\{ \sigma \in S_9 : \left\lceil \frac{i}{3} \right\rceil = \left\lceil \frac{j}{3} \right\rceil \Rightarrow \left\lceil \frac{\sigma(i)}{3} \right\rceil = \left\lceil \frac{\sigma(j)}{3} \right\rceil \right\}$$

The condition $\left\lceil \frac{i}{3} \right\rceil = \left\lceil \frac{j}{3} \right\rceil \Rightarrow \left\lceil \frac{\sigma(i)}{3} \right\rceil = \left\lceil \frac{\sigma(j)}{3} \right\rceil$ guarantees that only those row (column) permutations are allowed after which all the rows (columns) that were in the same band (stack) would be in a common band (stack).

So we could say that as a set, the sudoku group is the cartesian product $S_9 \times T \times T \times C_2$, but the definition of the group multiplication causes some difficulties. The natural approach that defines $(\pi_1, \sigma_1, \rho_1, l_1) \cdot (\pi_2, \sigma_2, \rho_2, l_2)$ to be $(\pi_1\pi_2, \sigma_1\sigma_2, \rho_1\rho_2, l_1l_2)$ fails because even the notion (π, σ, ρ, t) (where t is the generating element of C_2, so $t^2 = 1_{C_2}$) is not well-defined; it is not clear that if we take the action of this element of the sudoku group to a sudoku grid what transformation should be performed first, second, third and fourth.

The problem is that there are some transformations that do not commute with each other: taking the transpose of the grid and permuting the rows is not the same as performing these transformations in reverse order. However, the row operations and the column operations commute with each other, and it is also quite evident that the permutation of the digits is also interchangeable with any of the row or column operations. This means that the group generated by S_9 and the two copies of T is actually the direct product of the groups: $S_9 \times T \times T$.

In order to build in the transformation of taking transpose of a grid into the group we should first examine how can we make the group action on the set of the sudoku tables be well-defined?

Let us denote a sudoku grid by $[a_{i,j}]$, where the jth element of the ith row of the table is $a_{i,j}$. Then the relabelled grid with respect to a permutation $\pi \in S_9$ is $[\pi(a_{i,j})]$, the grid modified by the row and column operations $\sigma, \rho \in T$ is $[a_{\sigma^{-1}(i),\rho^{-1}(j)}]$. If all these transformations were performed the resulting grid would be $[\pi(a_{\sigma^{-1}(i),\rho^{-1}(j)})]$. If the transpose was not taken, the grid remains that. But if it was, we need to interchange the columns and the rows, so the sudoku grid on which $\pi \in S_9, \quad \sigma, \rho \in T$ and $t \in C_2$ was performed would be $[\pi(a_{\rho^{-1}(j),\sigma^{-1}(i)})]$. So if we defined the group multiplication on $S_9 \times T \times T \times C_2$ – which we have not yet done – the group action on the set of sudoku tables should be the following:

$$(\pi, \sigma, \rho, l) \cdot [a_{i,j}] = \begin{cases} [\pi(a_{\sigma^{-1}(i),\rho^{-1}(j)})], & if \quad l = 1_{C_2} \\ [\pi(a_{\rho^{-1}(j)\sigma^{-1}(i)})], & if \quad l = t \end{cases}$$

The first requirement of group action, that $1_G \cdot x = x$ is clearly satisfied this way and it is also clear that if we would like to multiply two elements from the sudoku group, we have no trouble with the elements of $S_9 \times T \times T$, since

$$((\pi_1, \sigma_1, \rho_1, 1_{C_2})(\pi_2, \sigma_2, \rho_2, 1_{C_2})) \cdot [a_{i,j}] = [(\pi_1 \circ \pi_2)(a_{(\sigma_1 \circ \sigma_2)^{-1}(i),(\rho_1 \circ \rho_2)^{-1}(j)})] =$$

$$= (\pi_1\pi_2, \sigma_1\sigma_2, \rho_1\rho_2, 1_{C_2})$$

The tool which solves this problem is called the semidirect product of groups, which is a generalization of the direct product.

Definition 2.6 *Let H and N be groups. Let $\varphi : H \to Aut(N)$ be a homomorphism and let us denote $\varphi(h) = \varphi_h$. Then the (outer) semidirect product of H and N with respect to*

φ is the group denoted by $N \rtimes_\varphi H$ which is the group on the set $N \times H$ equipped with the group multiplication $(n_1, h_1)(n_2, h_2) = (n_1\varphi_{h_1}(n_2), h_1h_2)$

It is a routine calculation to check that the defined structure is really a group. We also note here that if in the definition φ is the trivial homomorphism, which maps all elements of H to the identity automorphism of N, then we get the definition of the direct product of groups.

With this definition define the sudoku group as

$$S_9 \times T \times T \rtimes_\varphi C_2,$$

where $\varphi_t(\pi, \sigma, \rho) = (\pi, \rho, \sigma)$. With this definition the group action outlined above is evident.

Let us enumerate the elements of the sudoku group. S_9 has 9! elements, C_2 has two. We can calculate the number of elements of T as follows: $\sigma(1)$ can take 9 values – i.e. we have nine possibilities to choose for the place of the first column. But once $\sigma(1)$ is chosen, $\sigma(2)$ has only two possible value to take: those two with which the column $\sigma(1)$ stands in the same band. This two choices determine the value of $\sigma(3)$. Then we can start over, $\sigma(4)$ can take 6 values, $\sigma(5)$ has again 2 possibilities etc. Afterall we get that there are $9 \cdot 2 \cdot 1 \cdot 6 \cdot 2 \cdot 1 \cdot 3 \cdot 2 \cdot 1 = 1296$ elements in T.

Thus we can conclude that the sudoku group $S_9 \times T \times T \rtimes_\varphi C_2$ has

$$9! \cdot 1296 \cdot 1296 \cdot 2 = 1218998108160 \approx 1,2 \cdot 10^{12}$$

elements.

There has not yet been a purely mathematical calculation made to determine the number of fixed points for each element of the sudoku grid, however Ed Russell and Frazer Jarvis in 2006 [2] used a very efficient algorithm which made this calculation. After using Burnside's lemma the conclusion is that there are exactly

$$5472730538 \approx 5.5 \cdot 10^9$$

essentially different sudoku grids.

Chapter 3

Mathematical models and solving algorithms

3.1 Sudoku as an IP problem

The most obvious approach to solve a Sudoku puzzle could be modelling the problem as an integer programming problem.

To do so let us create the integer variables x_{ijk} for every $i, j, k = 1, 2, \ldots, 9$ where $x_{ijk} = 1$ means that the jth cell in the ith row must contain the digit k. Thus, we first have to ensure that for each pair (i, j), $x_{ijk} = 1$ can stand for exactly one k, so our first constraint is

$$\sum_{k=1}^{9} x_{ijk} = 1 \quad i, j = 1, \ldots, 9$$

for every i and j. At the same time this condition requires each cell to have a value.

Secondly, we have to ensure that each row and each column have to contain the digits from 1 to 9:

$$\sum_{j=1}^{9} x_{ijk} = 1 \quad i, k = 1, \ldots, 9$$

$$\sum_{i=1}^{9} x_{ijk} = 1 \quad j, k = 1, \ldots, 9$$

Finally, the condition which says that each 3×3 block must have each value from 1 to 9 exactly once is

$$\sum_{i=3n-2}^{3n} \sum_{j=3m-2}^{3m} x_{ijk} = 1, \quad k = 1, \dots 9 \quad n, m = 1, 2, 3$$

Finally, we let each variable to take two values, 0 or 1, which gives us the conditions

$$0 \leq x_{ijk} \leq 1 \quad i, j, k = 1, \dots 9$$

.

Since a wide range of LP solvers exist, which are also capable to solve IP problems, this model can be easily implemented. In my thesis I chose a python package called PuLP. It is available on the webpage `https://pypi.python.org/pypi/PuLP`. Trying to get familiar with the toolbox I found a solver program already installed with PuLP which is using the very same model, developed by Antony Phillips and Dr Stuart Mitchell. I used their implementation with little modification to write my own solver on a sudoku variant, Roku Doku, which will be explained later.

3.2 Sudoku as a graph coloring problem

There is a natural interpretation of the sudoku grid as a graph-coloring problem. Let us represent each cell of the sudoku grid by a vertex, and let us create an edge between two points if and only if the represented cells are in the same row, column or box. Let Sud_n be the graph representation of the $n^2 \times n^2$ sudoku grid, then Sud_3 is a 20-regular graph, since each cell has 8 box-neighbours, row-neighbours and column-neighbours, but there are 2-2 row- and column-neighbours which are already box-neighbours. Generally, Sud_n is $(3n^2 - 2n - 1)$-regular. Obviously an n-coloring of Sud_n gives a solution of the $n^2 \times n^2$ sudoku grid and vice versa.

Although this realization gives us the opportunity to use already existing graph coloring algorithms to solve sudoku puzzles, I would like to emphasize another question about the game which can be examined with this model. This was a new approach to the 17-clue problem by Agnes Herzberg and Ram Murty and it is discussed in detail in [3]. Here I demonstrate the main ideas of the article. To do so, let us recall some definitions and theorems of graph theory.

Theorem 3.1 *Let $G(V, E)$ be a simple graph. Let us denote by $p_G(k)$ the number of proper colorings of G by k colors. Then $p_G(k)$ is a polynomial of k.*

Proof: For an edge e of G let $G - e$ denote the graph which can be obtained from G by deleting the edge e; let G/e denote the graph which can be obtained by identifying the two endpoints of e (and deleting the multiple edges which may occur by the identification). We will use induction on the number of edges of G. If the graph contains no edges, then any vertex can get any of the k colors, so $p_G(k) = k^{|V|}$, which is obviously a polynomial of k. If G is any graph, then any proper k-coloring of G is a proper k-coloring of $G - e$. A k-coloring of $G - e$ is not a proper k-coloring of G if and only if the endpoints of e have the same color. The number of such k-colorings is $p_{G/e}(k)$, so $p_G(k) = p_{G-e}(k) - p_{G/e}(k)$. Both $G - e$ and G/e have fewer edges than G, thus the theorem is proved. □

In [3] a generalized version of this theorem was proved, namely:

Theorem 3.2 *Let $G(V, E)$ be a simple graph, and C a partial proper coloring of G using d_0 colors and t vertices. Let us denote $p_{G,C}(k)$ the number of proper completions of this partial coloring using $k \geq d_0$ colors. Then $p_{G,C}(k)$ is a polynomial of k.*

The proof is quite similar to the previous one and also another proof is discussed in [3] using partially ordered sets and Möbius functions.

Clearly, a sudoku puzzle can be represented by the graph Sud_3 (defined above) and a partial coloring C of Sud_3. The question is what conditions can guarantee $p_{Sud_3,C}(9) = 1$?

Another application of the chromatic polynomial of Sud_3 could be a new method for counting the number of possible Sudoku grids, which is the value of $p_{S_3}(9)$.

3.3 Sudoku and the exact hitting set problem

The exact hitting set problem is another NP-hard problem which can be used to model sudoku.

Definition 3.3 *Let S_1, S_2, ..., S_N be subsets of the set X. A subset X^* of X is said to be an exact hitting set if $|X^* \cap S_i| = 1$ for each $i = 1, \dots, N$ and for each $x \in X^*$ there exists i so that $x \in S_i$.*

In the case of the exact hitting set problem the task is to find such a subset X^* for a given set X and for its subsets $S_1, \dots, S_N$

The conversion of the sudoku to the exact hitting set problem uses a similar idea as the integer programming model. Let $X = \{x_{ijk}\}_{i,j,k=1}^{9}$ where x_{ijk} represents the case that the jth cell in the ith row is k. We choose some subsets of X in a way that an exact hitting set should be a solution. We will define four types of subsets:

1. In order to ensure that each cell have exactly one value we define $S_{ij} = \{x_{ijk}\}_{k=1}^{9}$ for each value of i and j in the range 1 to 9. So this means 81 subsets.

2. In order to ensure that each row contains each value exactly once we define $R_{ik} = \{x_{ijk}\}_{j=1}^{9}$ for each value of i and k in the range 1 to 9. So this also means 81 subsets.

3. In order to ensure that each column contains each value exactly once we define $C_{jk} = \{x_{ijk}\}_{i=1}^{9}$ for each value of i and k in the range 1 to 9, which type gives 81 subsets again.

4. And finally in order to ensure that each box to contains each value exactly once we define $B_{nmk} = \{x_{ijk} : 3n - 2 \leq i \leq 3n, 3m - 2 \leq j \leq 3m\}$ for each value of k in the range 1 to 9 and n, m for 1, 2 and 3, which are 81 subsets again.

It is easy to see that if X^* is an exact hitting set, then the meaning of $x_{ijk} \in X^*$ is that the jth element of the ith row is k. When we have a puzzle with some clues, we need to delete the conditions regarding the given cell: if the jth element of the ith row is k, then we do not need the sets S_{ij}, R_{ik}, C_{jk}, nor B_{nmk}, where n and m have their values from the set $\{1, 2, 3\}$ to satisfy the conditions $3n - 2 \leq i \leq 3n$ and $3m - 2 \leq j \leq 3m$.

There is a very effective algorithm from Donald Knuth to solve this problem, which is called the Algorithm X. This method can be effectively applied to the 9×9 Sudoku grid achieving one of the fastest solutions.

Chapter 4

Variants of Sudoku

4.1 Different size

As the sudoku became more and more famous and popular, other variants and generalizations were born. The most evident idea is to increase the size of the grid. As the original sudoku table is a 9×9 grid with nine 3×3 boxes we can generalize it without any effort. Taking an $n \in \mathbb{N}$ we need n^2 symbols, an $n^2 \times n^2$ grid with n separated $n \times n$ boxes. The rules remain the same: each row, column and box has to contain every symbol exactly once. The definitions that we made at the beginning of chapter 2 can be generalized without difficulties (blocks, stacks, bands). It has been proved that the completion of a partially filled sudoku grid (i.e. to solve puzzle) is an NP-complete problem as n increases [5].

It is easy to give a complete description of the case $n = 2$. We have already introduced the notion of essentially different sudoku grids, which can be generalized to different size without any problem. Let us recall the transformations from definition 2.2:

1. Relabelling.
2. Taking the transpose of the grid.
3. Exchanging two stacks.
4. Exchanging two bands.
5. Swapping two rows in a band.

6. Swapping two columns in a stack.

In connection with the number of equivalence classes we can state the following theorem.

Theorem 4.1 *There are exactly two essentially different* $2^2 \times 2^2$ *sudoku grids.*

Proof: Without loss of generality we can assume that the first block is in the canonical form, since we can obtain this situation by relabelling the symbols.

1	2		
3	4		

The third and fourth element in the first row must be 3 and 4, and the third and fourth element in the first column must be 2 and 4. Applying the 5. and 6. transformations on the second band and the second stack respectively we can make the grid look like the figure below:

1	2	3	4
3	4		
2			
4			

Now take a look at the block in the lower right corner. This box has to contain the number 4, but it can stand neither in the second row of the block, nor in the second column of the block. The only possible remaining cell is the diagonal cell in position (3,3). Then, we distinguish among three possibilities to continue completing our grid based on the three possible values of the lower right corner of the grid:

1	2	3	4
3	4		
2		4	
4			1

1	2	3	4
3	4		
2		4	
4			2

1	2	3	4
3	4		
2		4	
4			3

We leave it to the reader to figure out the solutions of the puzzles. The unique results for each puzzle are shown below:

1	2	3	4
3	4	1	2
2	1	4	3
4	3	2	1

1	2	3	4
3	4	2	1
2	1	4	3
4	3	1	2

1	2	3	4
3	4	1	2
2	3	4	1
4	1	2	3

The second and third tables are essentially the same, because we can obtain the second table from the third by taking the transpose and then swapping the symbols 2 and 3.

In order to finish the proof we have to show that the first and the second grids are essentially different. This means that we have to show that it is impossible to bring the two grids to the same form by applying some of the six transformation types. Let us do the following: we assign a graph with 4 vertices to the table where the vertices stand for the numbers 1, 2, 3, 4 respectively. If two numbers in any of the blocks of the table are standing in diagonally opposite positions we draw an edge between the two numbers. Thus, the edges of the first grid will be $1-4$ and $2-3$ only, and the edges of the second grid will be $1-4$, $2-3$, $2-4$ and $1-3$. The first graph has 2 edges, the second has 4. If we take a look at the six transformation types, we can see that the number of the edges of the graph remains the same. Indeed, for instance we can consider the relabelling transformation: any kind of relabelling can be represented by a permutation, and it is a well-known fact that every permutation can be written as a product of transpositions. (A transposition is a permutation which exchanges only two elements.) There are two types of swapping two numbers, which can affect the edges of the graph: if the swapped numbers are standing in a diagonally opposite position in every block, the swapping does not make any difference to the edges of the graph; if the swapped numbers are neighbours of each other, then the swapping deletes two edges, but creates two as well, so this leaves the number of edges invariant. It is trivial that the other 5 types of the transformations does not change this property either. As we noticed that these two graphs has different number of edges, we can conclude that the corresponding grids are essentially different. This completes the proof. □

The first part of the above proof, where we constructed the two essentially different grids, gives the idea of the enumeration of the $2^2 \times 2^2$ sudoku grids. Let us go through our reasoning again: If we keep the first block in the canonical form, we can see that there are $2 \cdot 2 = 4$ possible completions of the first row and the first column of the table, since we can change the order of 3 and 4 in the first row and also the order of the 2 and 4 in the first column. Since each of the second block of the first band and the second block of the first stack contains the value 4, column 3 or column 4 and also row 3 or row 4 will contain the symbol 4. This implies that we have the position of the symbol 4 in the last block uniquely determined. If we write it to its place, it can be seen that the value of the diagonally opposite cell in the last block, which is the intersection of the row and column which contains the symbol 4, can be any of the numbers 1, 2 or 3. Then, for each possibility there is a unique completion of the puzzle. Forgetting about the relabelling transformation we can conclude that there are $4! \cdot 2 \cdot 2 \cdot 3 = 288$ different $2^2 \times 2^2$ sudoku tables.

In view of these results the structure of the $2^2 \times 2^2$ sudoku grids seems to be quite simple, so we may be confident enough to solve the minimum-clue problem without any assistance of computer programs.

Theorem 4.2 *A $2^2 \times 2^2$ puzzle has to contain at least 4 clues to ensure a unique solution.*

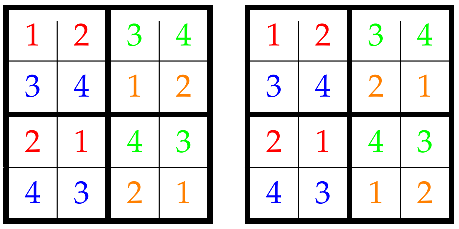
Proof: The following puzzle has 4 clues and has a unique solution:

1			
			2
		4	
	3		

1	2	3	4
3	4	1	2
2	1	4	3
4	3	2	1

There are no 2-clue puzzles, because in this case there would be at least two symbols that do not appear, thus their roles are interchangeable, which leads to at least two solutions. According to the previous theorem there are two essentially different sudoku grids. If we show that neither of them has a 3-clue puzzle, the theorem will be proved.

Let us have a look at the two representatives of the grids and consider the sets of cells which are marked by the same colors:



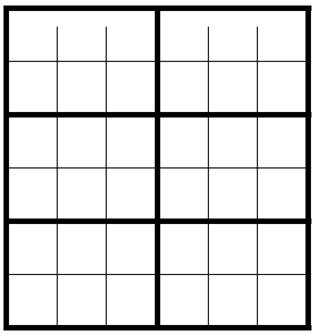
1	2	3	4
3	4	1	2
2	1	4	3
4	3	2	1

1	2	3	4
3	4	2	1
2	1	4	3
4	3	1	2

The sets marked with the same color are unavoidable sets in the following sense: Suppose that we have a puzzle, and its solution is the first table. If it does not have a clue from the red set, for instance, then the role of 2 and 1 would be interchangeable inside the red set without any impact on the other part of the grid. This implies that we need at least one clue from that region. The same argument holds for any of the other marked sets. Since there are 4 disjoint unavoidable sets, we need at least 4 clues. □

4.2 Different shape of the blocks

A different, but more general approach to broaden the concept of sudoku is changing the shape of the blocks - and also the size of the grid, if necessary. Let us first consider the case when the table is an $n \times n$ grid and the blocks are $r \times c$ rectangles. There should be c rectangles in each stack and r rectangles in each band, and thus $n = r \cdot c$. The figure shows the case of $n = 6, r = 2, c = 3$:



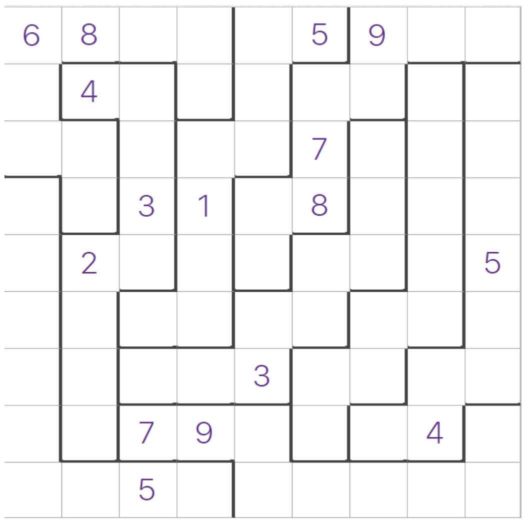
This variant is called Roku Doku (roku is the Japanese word for six). In general the collective name of sudoku variants that are smaller than the original are called Sub-Doku, or children sudoku. The latter expression is applied especially on the $2^2 \times 2^2$ sudoku.

Although in popularity none of these variants can compete with the original $3^2 \times 3^2$ sudoku, they can be used to get inspiration for the original one, or to test ideas, intuitions. Also they can popularize the game as beginners can gain sense of achievement easier when they try to solve smaller puzzles.

I used Roku Doku to write my own solver program based on the IP model that were discussed in section 3.1. The code can be found in the appendix.

Until this point we did not meet any variants that would require a new way of thinking, or at least new solving strategies, but why should we consider only these regular blocks, such as squares and rectangles? The idea of the 9×9 jigsaw sudoku is to generate nine connected regions inside the table, each region containing nine cells. The rules regarding to the columns and rows remain the same, and instead of square or rectangular blocks we have to put each symbol exactly once in each of the regions. An example puzzle is shown in figure 4.1. The solution 6.1 can be also found in the appendix.

Figure 4.1: Source: https://puzzlemadness.co.uk/jigsawsudoku/

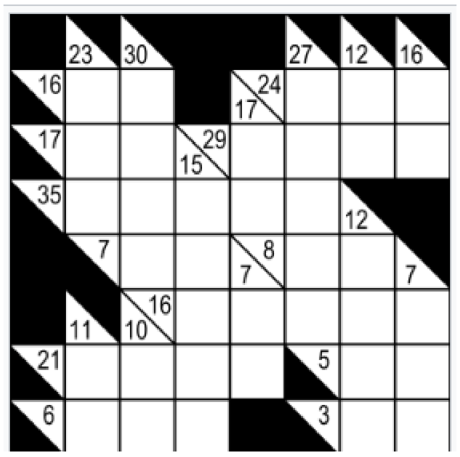


The mathematics behind these variants is analogous. All the solution models that were discussed earlier can be applied to these puzzles with obvious modifications. Also, the same ideas can be used to enumerate the possible grids, define equivalence classes, etc. However, the jigsaw sudoku may be complicated if we have the intention to change the shape of the regions. We will not discuss this possibility. Instead of that let us move forward to a more exciting variation of the game!

4.3 Killer Sudoku

Killer sudoku is a more complicated variation. This is a mixture of another game, kakuro with sudoku. Kakuro is similar to crossword, but instead of words it uses numbers. The picture 4.2 shows an example (whose solution 6.2 has been put into the appendix).

Figure 4.2: Source: https://hu.wikipedia.org/wiki/Kakuro



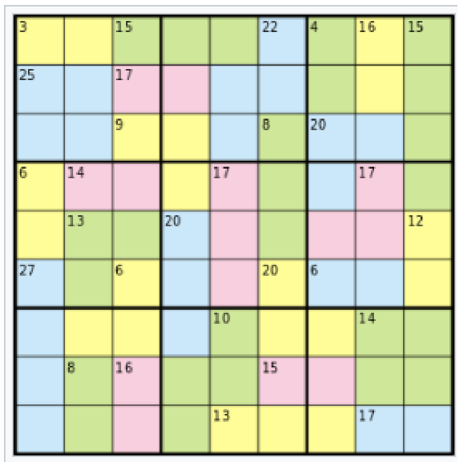
In kakuro the task is to fill the table with the numbers 1, 2, . . . , 9, while the given numbers show what the sum of the numbers has to be in the set of cells which they are assigned to. An additional constraint is that the partitions of the given numbers can contain a number only once. For example in the left bottom corner of the table 6 can not be obtained as $2 + 2 + 2$, nor $1 + 1 + 4$, so the only remaining combination is $1 + 2 + 3$, only the order has to be determined.

The amalgam of sudoku and kakuro is a sudoku grid which has additional regions (the border of these regions may cross the border of the blocks) and within these regions the sum of the numbers are specified. It is also a requirement that in each region every number occurs at most once. Figure 4.3 shows an example. Its solution can be found in the appendix 6.3.

As can be seen the only clues are the numbers which mark the sum of the regions. At first sight this game seems much more complicated than sudoku, but in some cases it is easier to crack a killer sudoku - though there are also harder ones, which are said to be requiring hours for professionals, too.

This game requires new logical strategies, but the same mathematical models

Figure 4.3: Source: https://en.wikipedia.org/wiki/killersudoku



can be applied to solve a puzzle like this. We will go through these briefly. Taking the IP model which was described in section 3.1 we just have to add the constraints for each region to have the required sum. First we recall that the binary variables x_{ijk} indicate that the jth element of the ith row is k. Let us denote the set of cells in the regions by $R_1, R_2, \ldots, R_s$ respectively, and the required sums in the regions $r_1, r_2, \ldots, r_s$. (for example in the case of the puzzle shown by figure 4.3, $R_1 = \{(1,1),(1,2)\}$ and $r_1 = 3$). The additional constraints which we need to include can be categorized in two types of categories. The first category contains the inequalities

$$\sum_{(i,j)\in R_t} x_{ijk} \leq 1$$

for all $t = 1, \ldots, s$ and $k = 1, \ldots, 9$.These coditions ensure that each region contains each value at most once. The second type of conditions are the equalities

$$\sum_{(i,j)\in R_t} \sum_{k=1}^{9} k \cdot x_{ijk} = r_t$$

for all $t = 1, \ldots, s$ to ensure the sum of the values being the number that is given in a region.

Since killer sudoku uses arithmetics (addition), I did not try to experiment with the graph coloring model, nor the exact hitting set problem. In the case of graph coloring algorithms we can ensure each cell to have different values in the regions, but the difficulties begin when we try to implement the condition on the sum of the numbers in certain regions. Since our numbers are represented by colors, this types of conditions cross the borders of this model. Similar problems occur in the case of the hitting sets.

Personally, I found this variation the most interesting. Though this game, in my opinion, would deserve more analysis, I have not found too much material which could answer at least a little part of my questions which came to my mind. Here are some of them:

What is a necessary and sufficient condition on the numbers assigned to the regions to ensure that the puzzle has a (unique) solution? A trivial necessary condition is that the sum of these numbers has to be $9 \cdot (1 + \cdots + 9) = 405$, since if we have a solution, then each number has to occur 9 times in the grid. But I can hardly believe that this would be a satisfactory condition, since the solvability of a killer sudoku puzzle may depend on the arrangement of the regions as well.

The question of the minimum clue problem has its own right here, too. What is the minimal number of the regions that must be given to ensure the puzzle to have a unique solution? This question with respect to this game seems much more complicated to me than in the case of the original sudoku.

Although purely mathematical theories about these questions would be extremely interesting, based on the earlier experiences about the original game I assume that it is more reasonable to approach these problems with the assistance of computer calculations.

Chapter 5

Summary

As I have mentioned in the introduction and as the title of my thesis suggests my goal was to analyze the mathematics behind sudoku. We have seen that this simple game has many connections to a wide range of fields of mathematics. We have connected it to Operations Research, Graph theory and Group theory. In some cases the number of possibilities made it hard to acquire results, but with the assistance of computers – and with some nice insights – these problems were still manageable.

There are other naturally arising topics which I did not cover in this writing, for example the problem of generating sudoku puzzles. These algorithms were rather empirical, trial and error approaches. The situation was similar to the classification of sudoku puzzles with respect to their difficulty. In this case most of the articles I read categorized the puzzles based on the strategies they required, which is quite subjective and depends on the puzzler.

I also did not mention logical strategies for the solution of the game. These are available on the internet in unlimited amount and they do not need much explanation.

All in all, I do hope that this thesis can provide an overall collection of the main mathematical approaches to the sudoku.

Chapter 6

Appendix

6.1 Solver program of Roku Doku

This code, which was written in python solves a 6×6 size sudoku puzzle which has 2×3 rectangular blocks. The program reads the puzzle from the "puzzle.txt" file, where the puzzle should be stored in a row-continuous format. The empty cells are represented by zeros.

The program uses the python package called PuLP to solve the IP problem representation of the sudoku. After the implementation of the problem the solution is written to the output and also stored in the "sudokuout.txt" file.

```
from pulp import *

def Plot_Sudoku(Sudoku):
    print("\n")
    for i in range(len(Sudoku)):
        print(Sudoku[i],end=' ')
        if i%3==2 and i%6!=5:
            print('|',end='')
        if i%6==5:
            print("\n",end='')
        if i%12==11:
            print('------------')
```

```
"""

"""
# Reading the puzzle
file = open("puzzle.txt","r")
Sudoku = []

for line in file:
    for i in line:
        Sudoku.append(int(i))

# Possible values are stored in the Digits vector
Digits = list(range(1,7))

# Index the columns and rows from 1 to 6
Row_index = Digits
Column_index = Digits
Values = Digits

#Creating of the Sudoku LP problem
prob = LpProblem("Sudoku problem", LpMinimize)

# Creating variables x_rcv: cth element of the rth row is v
choices = LpVariable.dicts("Choice", (Row_index, Column_index,Values)

# The arbitrary objective function is added
prob += 0, "Arbitrary Objective Function"

#Guarantee each cell to have only one value
for r in Row_index:
    for c in Column_index:
        prob += lpSum([choices[r][c][v] for v in Values]) == 1,""
```

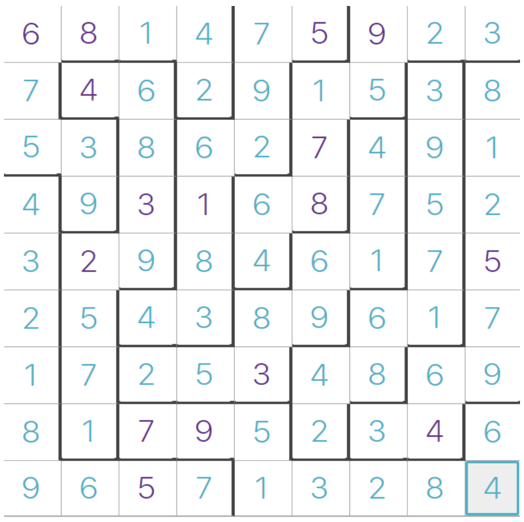
```
#Guarantee each row, column and box to contain each value exactly onc
for v in Values:
    for r in Row_index:
        prob += lpSum([choices[r][c][v] for c in Column_index]) == 1,

    for c in Column_index:
        prob += lpSum([choices[r][c][v] for r in Row_index]) == 1,""

    for x in [1,3,5]:
        for y in [1,4]:
            prob += lpSum([choices[x+i][y+j][v] for i in [0,1] for j

#Guarantee that the givens will show up in the appropriate cell - giv
#to the givens
for w in range(len(Sudoku)):
    if Sudoku[w] != 0:
        row_number = int( (w - w%6)/6) + 1
        column_number = w%6 + 1
        value = Sudoku[w]
        prob += choices[row_number][column_number][value] == 1,""

#Write problem to an .lp file
prob.writeLP("Sudoku.lp")

#Solving the problem
prob.solve()

Solution=[]

for r in Row_index:
    for c in Column_index:
        for v in Values:
            if choices[r][c][v].varValue == 1:
                Solution.append(v)
```

```
Plot_Sudoku(Sudoku)
Plot_Sudoku(Solution)

"""
# A file called sudokuout.txt is created/overwritten for writing to
sudokuout = open('sudokuout.txt','w')

# The solution is written to the sudokuout.txt file
for r in Row_index:
    if r == 1 or r == 3 or r == 5:
                    sudokuout.write("+————————+————————+\n")
    for c in Column_index:
        for v in Values:
            if choices[v][r][c].varValue==1:

                if c == 1 or c == 4:
                    sudokuout.write("|  ")

                sudokuout.write(str(v))

                if c == 6:
                    sudokuout.write("|\n")
sudokuout.write("+————————+————————+————————+")
sudokuout.close()
"""
```

6.2 Solution of puzzles

Solution of the jigsaw puzzle. (4.1)

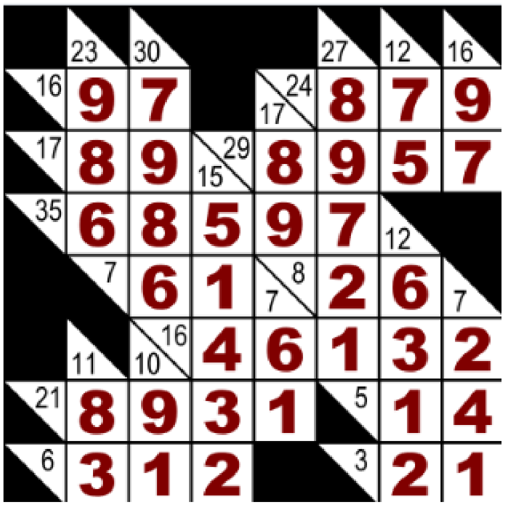
Figure 6.1



6	8	1	4	7	5	9	2	3
7	4	6	2	9	1	5	3	8
5	3	8	6	2	7	4	9	1
4	9	3	1	6	8	7	5	2
3	2	9	8	4	6	1	7	5
2	5	4	3	8	9	6	1	7
1	7	2	5	3	4	8	6	9
8	1	7	9	5	2	3	4	6
9	6	5	7	1	3	2	8	4

Solution of the kakuro. (4.2)

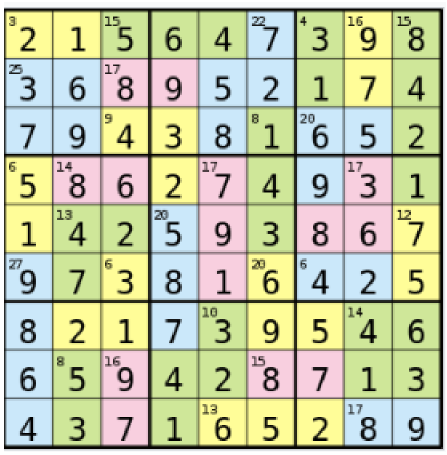
Figure 6.2: Source: https://hu.wikipedia.org/wiki/Kakuro



	23	30			27	12	16
16	9	7		24 17	8	7	9
17	8	9	29 15	8	9	5	7
35	6	8	5	9	7	12	
	7	6	1	8 7	2	6	7
	11	16 10	4	6	1	3	2
21	8	9	3	1	5	1	4
6	3	1	2		3	2	1

Solution of the killer sudoku. 4.3

Figure 6.3: Source: https://hu.wikipedia.org/wiki/killersudoku


3
15
22
4
16
15
25
17
9
8
20
6
14
17
17
13
20
12
27
6
20
6
10
14
8
16
15
13
17

2	1	5	6	4	7	3	9	8
3	6	8	9	5	2	1	7	4
7	9	4	3	8	1	6	5	2
5	8	6	2	7	4	9	3	1
1	4	2	5	9	3	8	6	7
9	7	3	8	1	6	4	2	5
8	2	1	7	3	9	5	4	6
6	5	9	4	2	8	7	1	3
4	3	7	1	6	5	2	8	9

Bibliography

[1] Bertram Felgenhauer and Frazer Jarvis. Mathematics of sudoku i. *Mathematical Spectrum*, pages 15–22, 2006.

[2] Frazer Jarvis and Ed Russel. Mathematics of sudoku ii. *Mathematical Spectrum*, pages 54–58, 2007.

[3] Agnes M. Herzberg and Ram Murty. Sudoku squares and chromatic polynomials. *Notices of American Mathematical Society*, 54:708–717, 06 2007.

[4] Gary McGuire, Bastian Tugemann, and Gilles Civario. There is no 16-clue sudoku: Solving the sudoku minimum number of clues problem via hitting set enumeration. *Experimental Mathematics*, 23:190–217, 2014.

[5] Takayuki YATO and Takahiro SETA. Complexity and completeness of finding another solution and its application to puzzles. E86-A, 05 2003.