

9 Sudoku modelling and solution

Sudoku problems are brain-teasers in which, given a partially-filled starting grid, an example of which is in Figure 5, the task is to fill the remaining small squares (referred to as **cells**) with the digits 1 to 9, whilst respecting the following **rules**

- Each row contains exactly one of each digit
- Each column contains exactly one of each digit
- Each 3×3 square contains exactly one of each digit

As set out below, the study of Sudokus provides a good example of modelling and solution using linear programming (LP) and integer programming (IP), as well as an insight into some more advanced techniques in optimization, complexity theory and general scientific method.

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

Figure 5: Sudoku starting grid

9.1 Some definitions and mathematical properties

Figure 5 illustrates the popular 9×9 problem. This is the case $n = 3$ of the general $n^2 \times n^2$ problem. The problem in Figure 5 has a unique solution and, as such, is said to be a **proper Sudoku**. Here are some (proved) mathematical results about Sudokus for $n = 3$

- The fewest possible starting values for a proper Sudoku is 17.
- The number of essentially different proper Sudoku problems, “when symmetries such as rotation, reflection, permutation, and relabelling are taken into account”, is 5,472,730,538.

The problem of solving general $n^2 \times n^2$ Sudokus is known to be NP-complete. This means that for solving general problems, there can exist no algorithm whose running time is a polynomial in n . Unless stated otherwise, the discussion below relates to the classical 9×9 Sudoku.

9.2 Modelling and solution using linear and integer programming

Sudokus provide a good example of modelling and solution using linear and integer programming. The key when modelling using linear and integer programming is clear identification of the decision variables to be used to model the unknowns. In the case of Sudokus, it’s tempting to use $9^2 = 81$ decision variables, one for each cell, in which the variable x_{ij} models the decision “What digit goes in square (i, j) ?”. Unfortunately, it is then not possible to model the Sudoku rules using such a set of variables. An alternative is to use $9^3 = 729$ decision variables x_{ijk} for all $i = 1, 2, \dots, 9$; $j = 1, 2, \dots, 9$ and $k = 1, 2, \dots, 9$, in which x_{ijk} models the decision of whether or not to put digit k into the (i, j) cell. As a consequence, each decision variable must take the value 0 or 1.

Generally when modelling linear and integer programming problems there is a linear function of the decision variables that measures the value of the decision, so that the best of all the feasible assignments of decision variables can be identified. There is no such optimality measure for Sudokus: the goal is simply to find a feasible assignment of values for the decision variables. As such, the Sudoku model is an example of a **feasibility problem**. The linear objective function $\sum_{i=1}^9 \sum_{j=1}^9 \sum_{k=1}^9 c_{ijk} x_{ijk}$ is trivially defined to be identically zero by setting all values of c_{ijk} to zero.

There are four sets of constraints that must be defined to capture the Sudoku rules. Firstly, in each cell (i, j) exactly one digit must appear. This is modelled by requiring that the total number of digits

appearing in cell (i, j) must sum to 1 (since each decision variable must take the value 0 or 1). Hence, for all rows $i = 1, 2, \dots, 9$ and columns $j = 1, 2, \dots, 9$

$$\sum_{k=1}^9 x_{ijk} = 1$$

The remaining constraints correspond to the three standard Sudoku rules.

- Each row contains exactly one of each digit.

Thus, for all columns $i = 1, 2, \dots, 9$ and digits $k = 1, 2, \dots, 9$,
$$\sum_{j=1}^9 x_{ijk} = 1$$

- Each column contains exactly one of each digit.

Thus, for all rows $j = 1, 2, \dots, 9$ and digits $k = 1, 2, \dots, 9$,
$$\sum_{i=1}^9 x_{ijk} = 1$$

- Each 3×3 square contains exactly one of each digit.

This is harder to express, and can be done by defining the set of row and column indices that define the top-left-hand corner of 3×3 squares in the grid. Let this set be $V = \{1, 4, 7\}$. The remaining cells in a 3×3 square are given by offsets of 0, 1 or 2 from the top-left-hand corner, yielding the following set of constraints.

For all columns $i \in V$ rows, $j \in V$ and digits $k = 1, 2, \dots, 9$,
$$\sum_{r=0}^2 \sum_{c=0}^2 x_{i+r, j+c, k} = 1$$

Note that there are 81 individual constraints in each of the four sets, a total of $4 \times 81 = 324$. Each of the constraints defines that a sum of exactly 9 decision variables must equal 1. Each decision variable appears in exactly four constraints.

Mathematically, the decision variables correspond to $\mathbf{x} \in \mathbb{R}^{729}$ and, when coefficients of the constraints are accumulated into a matrix $A \in \mathbb{R}^{324 \times 729}$ its nonzero pattern is illustrated in Figure 6. Only $9 \times 324 = 2916$ of the $729 \times 324 = 236,196$ entries in the matrix are nonzero, a density of 1.23%. Although the Sudoku model is small in the context of practical LP and IP problems (whose dimensions may be in the millions) avoiding computing with zeros is essential to the efficient solution of large LP and IP problems. This is referred to as **exploiting sparsity**.

The constraints form a system of equations $A\mathbf{x} = \mathbf{b}$, where $\mathbf{b}$ is a vector of ones. In general, a system of 324 equations and 729 variables will have an infinite number of solutions (forming a hyper-plane in $\mathbb{R}^{729}$ of dimension $729 - 324 = 405$). This will contain a very large number of points with values of 0 or 1, each of which corresponds to a possible completion of a blank Sudoku grid. However, what defines a specific Sudoku problem is the set of starting values. These are used to fix the values of a subset of the decision variables. Specifically, if the starting grid contains a digit k in cell (i, j) , then the model has the constraint $x_{ijk} = 1$. Remarkably, if the starting grid contains the minimal 17 digits required for a proper Sudoku, the set of feasible solutions to the system of equations shrinks to a region containing just one point with values 0 or 1: the unique solution of a proper Sudoku.

When the model formulated above is communicated to a decent integer programming solver (such as HiGHS), it will identify the solution in very little time. Reasons for this are explored in Sections 9.3 and 9.4.

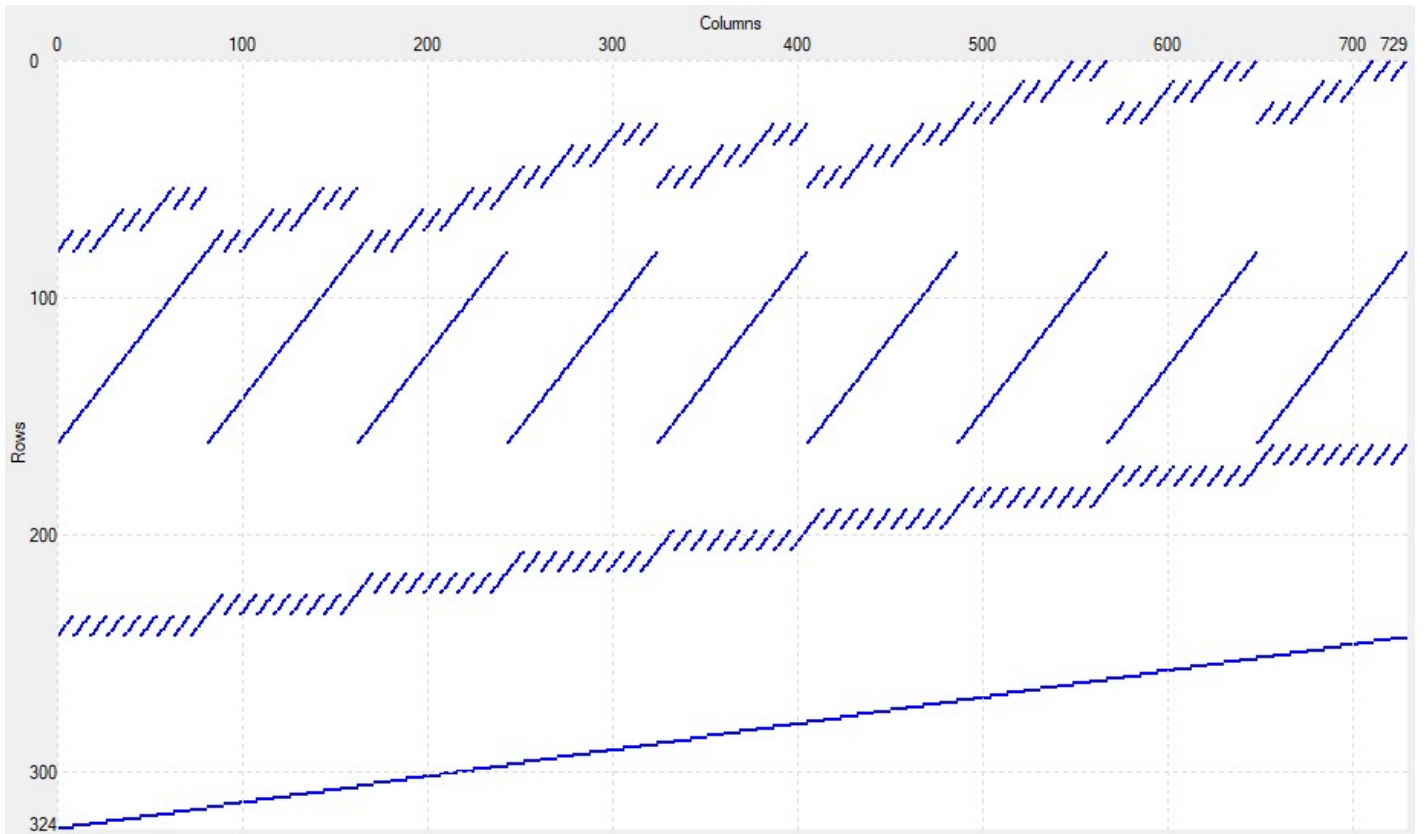


Figure 6: Sudoku constraint matrix

9.3 Solving Sudokus by LP

When experimenting with a small number of grids for proper Sudokus, I noticed that the constraint that each decision variable must take the value 0 or 1 ($x_{ijk} \in \{0, 1\}$) was unnecessary: it was sufficient to restrict x_{ijk} to the interval $[0, 1]$, converting the model from IP to LP. In Summer 2022 I supervised an Operational Research MSc student on a dissertation project whose aim was to determine whether the conjecture that all proper Sudokus can be solved as LPs was true.

When attempting to prove something, it is worth looking for a counter-example to avoid the wasted effort of trying to prove a result that is false. In this case, if just one proper Sudoku gave a fractional solution when solved as an LP, then the conjecture would be false. To search for such a counter-example, the student used an existing data set of a million proper Sudokus. She found that each could be solved as an LP! Whilst not a proof, this was reason for optimism that the result might be true. That said, if all 5,472,730,538 distinct proper Sudokus could be solved as LPs, this *would* constitute a proof. Although five thousand times larger than the data set that she used, such an exhaustive search would be practical computationally.

The student then found another data set of 4000 proper Sudoku puzzles categorised by how difficult they would be for humans to solve: easy, medium, hard or diabolical, with 1000 of each. When solved as LPs, 306 of the hard problems and 732 of the diabolical problems had fractional solutions. Hence the conjecture was false (for $n = 3$).

Scientifically, the moral of this story is that it's more important that a data set is good, than big!

The smallest non-trivial Sudokus are the case $n = 2$, for which the grid consists of four 2×2 squares, in which exactly one of the digits 1 to 4 must be assigned to each cell whilst respecting the standard Sudoku rules. There are only 288 distinct proper Sudokus for $n = 2$ so, since the set of starting grids was available, it was simple to test them all. It was identified that all could be solved as LPs. Hence the conjecture is true for $n = 2$.

Note that, since the general $n^2 \times n^2$ Sudoku problem is known to be NP-complete, it cannot be solved

in polynomial time. Hence, although the conjecture is true for $n = 2$, it was bound to be false for some value of n , since LPs can be solved in polynomial time. Sudokus for any value of n can be modelled and solved as IP problems, corresponding to the fact that the solution of general IP problems is also NP-complete.

9.4 Solving Sudokus by presolve

When Sudoku decision variables are fixed by values in the starting grid, it follows that the values of other decision variables will be fixed as a consequence. For example, the 5 in the $(1, 1)$ cell in the starting grid of Figure 5 fixes $x_{115} = 1$. However, to a human, it is obvious that no other digit can be in the $(1, 1)$ cell, so this fixes $x_{11k} = 0$ for $k = 1, 2, 3, 4, 6, 7, 8, 9$. Decision variables whose values have been fixed can be removed from the problem. Further, the constraint

$$x_{111} + x_{112} + x_{113} + x_{114} + x_{115} + x_{116} + x_{117} + x_{118} + x_{119} = 1 \quad (1)$$

on the number of digits that can be present in the $(1, 1)$ cell can also be removed from the problem before it is solved computationally.

Although mathematical simplification is to be encouraged, in the case of modelling, it is best left to a computer. Before optimization software solves an LP or IP, a procedure known as **presolve** reduces the size of the problem to be solved by applying rules recursively to eliminate variables and constraints. Once the reduced problem has been solved, a **postsolve** procedure deduces the optimal solution of the original problem from the optimal solution of the reduced problem by working backwards through the sequence of reductions.

Although presolve is a complex process, the Sudoku model allows an insight into a few of its features to be gained. For instance, how can the elimination (above) of x_{11k} for $k = 1, 2, \dots, 9$ be identified algorithmically? Firstly, presolve will look for fixed variables, and remove them from the problem at the fixed value. So, it removes x_{115} with a value of 1. This will modify the equations in which x_{115} occurs. In particular, (1) will become

$$x_{111} + x_{112} + x_{113} + x_{114} + x_{116} + x_{117} + x_{118} + x_{119} = 0 \quad (2)$$

Now, since each of the variables in (2) must take a value in $[0, 1]$, the value of the LHS of the equation lies in $[0, 8]$, and can only be zero if all of the variables have the value 0. This **forcing row** fixes the value for all of the variables in the equation, and the equation can be removed from the problem.

The variable x_{115} also appears in the following equation ensuring that there is exactly one 5 in row 1.

$$x_{115} + x_{125} + x_{135} + x_{145} + x_{155} + x_{165} + x_{175} + x_{185} + x_{195} = 1 \quad (3)$$

After fixing $x_{115} = 1$, (3) becomes

$$x_{125} + x_{135} + x_{145} + x_{155} + x_{165} + x_{175} + x_{185} + x_{195} = 0,$$

another forcing row that allows its variables to be fixed at zero. This corresponds to the logical deduction that, since there is a 5 in the $(1, 1)$ cell, there can be no 5 in any other cell in row 1.

As variables are successively fixed and removed from the problem, along with equations corresponding to forcing rows, other forcing rows may be created, leading to further reductions. Those used to solving easy Sudokus may anticipate that this process will continue until all the variables have been fixed, and all the equations removed. In presolve this is referred to as “reducing the problem to empty”. When this situation is reached, the optimal solution of the original problem is known, since all decision variables have been fixed to values during presolve.

For harder Sudokus, once a human has made all the easy deductions, the situation may arise where a particular cell must contain exactly one of two digits. Suppose that this is the cell (i, j) , and that the

digits that it may contain (without loss of generality) are 1 and 2. In presolve, the following equation will appear in the reduced problem

$$x_{ij1} + x_{ij2} = 1 \quad (4)$$

Such a **doubleton equation** can be used to eliminate one of the variables. The choice is arbitrary, but suppose

$$x_{ij2} = 1 - x_{ij1}$$

is chosen, allowing x_{ij2} to be eliminated from any other equations in which it appears, and allowing (4) to be eliminated from the problem. With such eliminations it is possible that, ultimately, an equation

$$x_{ij1} = 1$$

is found. This fixes x_{ij1} to 1, allowing further reductions. When a human solves a Sudoku, the logical consequences of choosing to put a 1 or 2 in cell (i, j) are followed until either the Sudoku is complete, or a contradiction is found. In the latter case, the digit chosen earlier is deduced to have been incorrect. In presolve, the two alternatives are “programmed in” algebraically. In postsolve (which reverses the sequence of presolve reductions), when the fixed value $x_{ij1} = 1$ is substituted into (4), the optimal value $x_{ij2} = 0$ is deduced.

Even for harder Sudokus, experiments showed that many of the LPs were reduced to empty, ensuring that an integer-valued solution to the LP was obtained. Indeed, the student interpreted another human Sudoku solution technique in terms of a further presolve rule. When presolve did not reduce the LP to empty, it was possible that the feasible region of the reduced problem contained more than just a single point, so could have multiple vertices. [Note that the feasible region cannot contain more than one point where all values are 0 or 1, since that would imply that the original Sudoku was not proper.] Since the objective function is zero, the simplex algorithm will terminate at the first feasible vertex that it finds, and this may be a point with have fractional values, resulting in a non-binary solution to the original LP after postsolve.

9.5 Tight and weak MIP formulations

The MIP formulation above is said to be tight, in that some of the constraints can be relaxed, and the solution of the problem is still the solution of the Sudoku. For example, the constraints on the number of digits in each cell (i, j) ,

$$\sum_{k=1}^9 x_{ijk} = 1$$

can be relaxed to

$$\sum_{k=1}^9 x_{ijk} \leq 1$$

since all of the variables x_{ijk} for $j = 1, 2, \dots, 9$ cannot be zero, as this would imply that there is no digit in cell (i, j) . Without a digit in this cell, some of the other equality constraints - that there is exactly one of each digit in each row, column and 3×3 square - would not be satisfied.

If, in addition, the constraints that there is exactly one of each digit in each row

$$\sum_{j=1}^9 x_{ijk} = 1$$

are relaxed to

$$\sum_{j=1}^9 x_{ijk} \leq 1$$

then, all of the variables x_{ijk} for $j = 1, 2, \dots, 9$ cannot be zero, as this would imply that digit k did not appear in row i . Hence some of the other equality constraints - that there is exactly one of digit k in each column and 3×3 square - would not be satisfied.

If, in addition, the constraints that there is exactly one of each digit in each column

$$\sum_{i=1}^9 x_{ijk} = 1$$

are relaxed to

$$\sum_{i=1}^9 x_{ijk} \leq 1$$

then, all of the variables x_{ijk} for $i = 1, 2, \dots, 9$ cannot be zero, as this would imply that digit k did not appear in column j . Hence one equality constraint - that there is exactly one of digit k in a particular 3×3 square - would not be satisfied.

However, the constraints that there is exactly one of each digit in each 3×3 square

$$\sum_{r=0}^2 \sum_{c=0}^2 x_{i+r, j+c, k} = 1$$

cannot also be relaxed, as all the equality constraints in the original formulation would be relaxed, so the starting grid would be a feasible solution. By symmetry, at least one of the four classes of equality constraints must remain if the Sudoku is to be solved as a *feasibility* problem.

9.5.1 A weaker MIP formulation

As observed above, if all four classes of equality constraint are relaxed, the starting grid is a feasible solution. However, the Sudoku is solved if the MIP is given the objective of maximizing the sum of variables. Now completing the Sudoku is the optimal thing to do!

9.5.2 The weakest MIP formulation?

An even weaker formulation does not put lower bounds of 1 on the variables corresponding to the starting grid, but merely requires them to sum to at least the number of digits in the starting grid. However, with each variable being in $[0, 1]$, this constraint is only satisfied when they are all equal to 1. Hence presolve uses this forcing row to fix the variables to 1.

9.5.3 Results for the hardest Sudoku

What is claimed to be the hardest sudoku puzzle is illustrated in Figure 7, and was created by a Finnish mathematician, Arto Inkala. Using HiGHS, it must be solved as a MIP, and the size of the problem after presolve, together with the solution time, are given in Table 1. It is seen that as the formulation is weakened, the size of the presolved problem grows, so the solution time increases. That said, for the weakest formulation, presolve immediately fixes the starting digits, so the reduced problem and solution time are identical. This illustrates the truism that a MIP formulation should be as tight as possible, without introducing an excessive number of constraints. The interest in the weak formulation is discussed in Section 9.5.4.

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

Figure 7: The hardest Sudoku

Formulation	After presolve		Time (ms)
	Variables	Constraints	
Tight: Four sets of equations	166	158	60
Weak: Three sets of inequalities	182	185	920
Weaker: Four sets of inequalities and an objective	223	224	1250
Weakest: Four sets of inequalities, an objective and constraint fixing start values	223	224	1250

Table 1: The hardest Sudoku: results applying HiGHS to the MIP formulations

9.5.4 Finding the best solution to an infeasible Sudoku

If there is no solution for a Sudoku problem, there is some interest in finding the maximum number of digits that can be defined. So long as the initial starting grid does not violate any of the Sudoku rules, it is a feasible solution for the weakest formulation. Since the objective is to maximize the number of digits placed in the grid, a MIP solver will find the “best” solution to an infeasible Sudoku.