A Formal Mathematical Framework for Sudoku

5/25/2025

Introduction

Sudoku, a logic puzzle played on a 9×9 grid, appears simple at first glance. Yet beneath the surface lies a structure rich in combinatorics, constraint satisfaction, and algorithmic complexity. This paper formalizes Sudoku as a logical system, beginning with axioms and then building theorems and deductive strategies that a solver (human or computer) can use to efficiently and correctly complete any valid puzzle. This paper was written to prove to my friend M.J. that the reason I perform poorly in this game lies purely in its complex mathematics, and not in my own (limited and unskillful) gameplay :).

Comment on Intellectual Property

I add this section solely for the purpose of posting this work. All the work in this paper is my attempt to discover the mathematics behind the game Sudoku. Some theorem names and concepts (e.g., the Naked Pair Theorem) are pre-existing, and not my ideas. All proof attempts are my unaltered, unaided work(so don't judge me if they're bad :)). Everything I based this paper on is found in the two links in Theorem 2.6. This is purely a compilation of mathematical research combined with my own proofs and ideas.

Axioms of Sudoku

In simplistic terms, a Sudoku board is just an application of a Latin square. A Latin square is a $n \times n$ grid of cells $C_{(i,j)}$, each of which takes on an integer value $\in \{1, \ldots, n\}$. This is a branch of mathematics which has no current direct equation to solve it, rather, it can be solved systematically using constraint logic, integer programming, or group theory.
Already, we are off to a promising start in my defense for the complexity of this game :)

The Sudoku Function: Let C be a 9×9 grid of cells indexed $C_{(i,j)}$, where $i, j \in \{1, \ldots, 9\}$. The Sudoku Function, denoted *f*, is the following: f: $C_{(i,j)} \to \mathbb{Z}$. A solution to Sudoku is a bijective mapping $C_{(i,j)} \to \{1, \ldots, 9\}$ satisfying the following axioms:

- (S1): Each row contains the digits 1–9 exactly once.
- (S2): Each column contains the digits 1–9 exactly once.
- (S3): Each 3×3 subgrid (region) contains the digits 1–9 exactly once.
- (S4): The initial clues are fixed and unchangeable.

Theorems and Corollaries

Common Variables and their properties:

$i, j \in \mathbb{Z} \to$ refers to the position of a cell on the Sudoku board at location(i, j).
$C_{(i,j)} \to$ refers to cell at row i, column j

Theorem 1.1 (Uniqueness Theorem): Each digit can appear only once per row, column, and region.

Proof. Direct from (S1)–(S3). □

Theorem 1.2 (Candidate Elimination): If a digit appears in a cell's row, column, or region, it is not a valid candidate for that cell.

Proof. Suppose the function $f : C_{(i,j)} \to \{\text{all } x : x \text{ satisfies S1-S3}\}$ which maps each cell to all possible candidate numbers for each $C_{(i,j)}$. Now suppose digit d is placed elsewhere in the same row/column/box as cell $C_{(i,j)}$. Then by (S1)–(S3), $d \notin f\left(C_{(i,j)}\right)$. □

Corollary 1.2.1 (Singleton Rule): If a cell has only one valid candidate, that digit must be placed in that cell.

Corollary 1.2.2 (Hidden Singleton Rule): If a candidate appears only once in the candidate list of a row/column/region, it must go there.

Corollary 1.2.3 (Single Remainder Rule): If every cell in a row, column or box has a value except for one cell, then it only has one valid candidate, which, by Corollary 1.2.1, must be its value.

Theorem 1.3 (The Summation Theorem): $\forall$ cell $C_{(i,j)} \in C$, $\sum_{i=1}^{9}\sum_{j=1}^{9} C_{(i,j)} = 405$.

Proof. This is a standard mathematical induction proof, the basic outline is provided. We know that the sum of each cell's value in a row is $\sum_{x=1}^{9} x$, and $\sum_{x=1}^{n} x = \frac{n(n+1)}{2}\ \forall\ n \in \mathbb{N}$ (proof by mathematical induction) $\implies \sum_{i=1}^{9} i = \frac{9(9+1)}{2} = 45$. Since there are 9 columns, and each column sums to 45, thus the total sum of every cell on the Sudoku board is $\sum_{j=1}^{9} 45 = 9 \cdot 45 = 405$ □

Theorem 1.4 (Candidate Existence Theorem):
This last theorem proposes an equation that performs a very crucial task in solving Sudoku Latin Squares.

Suppose the sequence of solved tuples denoted $S_{k=1}^{l} = \{(i_k, j_k, d_k)\}_{k=1}^{l}$. Where: l is the length of the sequence, or the total number of known cells, and (i, j) is the location of the k^{th} cell with the value d_k

Now suppose the function $t : (S_{k=1}^{l}, C_{(i,j)}) \to \{x : x \notin (R_i \cup C_j \cup B_{(i,j)}\}$ where R_i is the set of values in row i, C_j is the set of values in column j, and $B_{(i,j)}$ is the set of values used in the 3×3 box containing $C_{(i,j)}$. This function is vital to future theorems, proofs and algorithms, and as such, we give it a special name, called the *Tham Function*, taking the parameter $S_{k=1}^{l}$, called the *Khun Candidate List (K.C.L.)*, and the current cell at location (i, j).

Then, $\forall C_{(i,j)} \in C, \exists$ a non-empty set of possible values $P = \{x : x \in\ \ t\ (S_{k=1}^{h}, C)\}$. In layman's terms, for every unsolved cell $C_{(i,j)}$ (that is, the ones not in the K.C.L.), there exists a **non-empty** list of integers such that each integer in the list is a possible value of the cell that does **not** violate any of the outlined axioms.

Proof. This proof is quite simple, the function t is a well-defined function, with the output being simply the set

$$t(S_{k=1}^{l}, C_{(i,j)}) = \{x \in \{1, \ldots, 9\} : x \notin R_i \cup C_j \cup B_{(i,j)}\}$$

This is a well-defined set because:

- R_i is all values in cell $C_{(i,j)}$'s row, C_j is the same for the column, and $B_{(i,j)}$ is the values in the cells 3×3 sub-box.

  Note[1] *each of R_i, C_j, and $B_{(i,j)}$ are all $\subset S_{k=1}^{l}$, and are all constraints specific to cell $C_{(i,j)}$. The union $R_i \cup C_j \cup B_{(i,j)} \subseteq S_{k=1}^{l}$ represent all viable solutions specific to that cell.*

  Note[2] *this set can never be empty (a requirement outlined before)(proven in lemma 1.4.1 below).*

- The union of these sets represents all digits that are **not allowed** in cell $C_{(i,j)}$, due to constraints (S1)–(S3).

Then, by definition of set difference, our *Tham Function* from before is:

$$t(S^l_{k=1}, C_{(i,j)}) = \{1, \ldots, 9\} \setminus (R_i \cup C_j \cup B_{(i,j)})$$

This set always exists, therefore, for every cell $C_{(i,j)} \in C$, there exists a set of candidate values that satisfy the Sudoku axioms S1-S3, relative to the current configuration $S^l_{k=1}$. The only case where this set would **not** exist is when an incorrect cell value exists in the K.C.L., that is: This result only holds under the assumption that the current state $S^l_{k=1}$ is consistent with axioms (S1)–(S3)

□

Lemma 1.4.1:

Proof. If there is an empty square, and every value is the K.C.L. is correct, then $\forall$ cells $C_{(i,j)} \in C, \exists$ **at least one** $x : x$ satisfies S1-S3 for the cell $C_{(i,j)}$. (Notice how this is different from the function given in Theorem 1.2, or the Tham Function, which claims a set of potential candidate values)
Each of $R_i, C_j, B_{(i,j)}$ can contain at most 8 digits, since the current cell is excluded. As such, in order for each number to be used exactly once, then the remaining value is uniquely determined by the unused digit. Since $C_{(i,j)}$ has only one solution, by Corollary 1.2.3, it must be that cells value. Even if multiple candidates are possible, the lemma still holds by existence

□

Intermediate Logical Deductions

Theorem 2.1 (Naked Pair Theorem): If two cells in the same unit (row/col/box) have exactly the same pair of candidates, those two digits can be removed from all other cells in that unit.

Proof. Let S_{i1}, S_{i2} be two cells in the same unit with candidates {a, b}. Since each of a, b must be used once and only once in the unit, and no other cells can contain them without violating uniqueness, remove a, b from all other cells in that unit. □

Theorem 2.2: (General Naked Pair Theorem): If $n \in \{1, \ldots, 8\}$ cells in the same unit all possess the exact same set of n candidates, then those n digits can be removed from all other cells in that unit.

Proof. Same proof as Theorem 2.1, this theorem does not work for $n = 9$, since when n reaches 9, no information can be deduced from this theorem. □

Theorem 2.3 (Inverse Naked Pair Theorem): If 2 cells have a list of suitable candidates, but they have a matching pair of candidates that do not appear anywhere else in the unit, then all other candidates can be removed.

Proof. Suppose two cells C_1 & C_2 have the corresponding set of possible candidates L_1 & L_2. If 2 numbers x and $y \in L_1$ and L_2 and do not appear anywhere else in the unit, by the axioms S1-S3, x or y must go in either one of those two locations. □

Theorem 2.4 General Inverse Naked Pair Theorem: Following the same logic and structure as Theorem 2.2, the General Inverse Naked Pair Theorem works for numbers greater than 2, but less than 9.

Theorem 2.5 (Number of Valid Sudoku Grids): The number of valid, completed 9×9 Sudoku grids is:

6,670,903,752,021,072,936,960

Theorem 2.6 (Minimum Clues for Unique Solution): A Sudoku puzzle must have at least 17 clues to admit a unique solution.

Proof. Both Theorem 2.4 & 2.5 are proven in these first(2.3) and second(2.4) papers(see full citations below). It is beyond anything I have the ability to compute or prove as I don't own a NASA supercomputer. □

The Holden Algorithm

Christened after our G.O.A.T. Tyler Holden (iykyk), the Holden Algorithm is a mathematically adapted method to solving Sudoku games.

Step 1: The first values in the Khun Candidate List(K.C.L.) are the starting hints that are given upon starting the game. We then can use the Tham Function with parameters of the K.C.L. and all cells $C_{(i,j)} \in C$.

Step 2: We now have the output of the Tham Function, which is all possible values for cell $C_{(i,j)}$. This can be stored in a dictionary $V = \{C_{(i,j)} : \ t(S^l_{k=1}, C_{(i,j)})\} \forall C_{(i,j)} \in C \notin S^l_{k=1}$.

Step 3: We select the cell $C_{(i,j)}$ with the smallest number of candidates. If the length of the output of the Tham Function is 1, then we know that it is the only correct answer, thus it is added to the K.C.L.

Step 4: Check for all cases of the intermediate logical deductions to reduce the output of the Tham Function for all possible cells, if a cell is reduced to 1 possible candidate, then assign it and put it in the K.C.L.

Step 5: If the length of the shortest output of the Tham Function is greater than 1, then select the cell with the shortest output. Assign it the value that appears least frequently among all Tham Function outputs in its unit. Flag this as a chosen value, as should you get to a point where the board is unsolvable, you go back to this cell and choose the next in line value. This introduces backtracking. If a contradiction is reached, revert to this cell and try the next value in its candidate list.

Rinse and Repeat

The complex mathematics behind Latin Squares is complex yet fascinating. Maybe, in the not so distant future, I would put the time into actually get good at Sudoku games rather than writing math papers about them, but until that day, THESE THINGS ARE HARD M.J. LOOK AT THE MATH. Though overall I quite enjoyed doing this analysis, 8.5/10.

References

- Tjusila, G.; Besan¸con, M.; Turner, M.; and Koch, T. 2024. How many clues to give? A bilevel formulation for the minimum Sudoku clue problem. Oper. Res. Lett., 54: Paper No. 107105, 6.
- S.F.Bammel, J.Rothstein, The number of 9×9 Latin squares, Discrete Mathematics 11 (1975) 93–95
- All general information about Sudoku (strategies, Theorem concepts & names) were found at foxy-sudoku.com.