Boolean Satisfiability & Sudoku

Alexei E. Angelides

Our goal in what follows is to show that:

Theorem: There is a truth-value assignment for a set of Boolean propositions *if and only if* there is a solution to Sudoku.

The idea behind the theorem is that it is possible to "reduce" combinatorial problems, such as finding a solution to Sudoku puzzles, to finding a truth-value assignment for the set of propositions that describes those puzzles. Conversely, we might think of the theorem as asserting that it is also possible to reduce a set of propositions in Boolean logic to a solution to a combinatorial problem, again like Sudoku. The biconditional (*iff*) in the theorem asserts that these two tasks are one and the same.

For this proof, we first need to describe the conditions defining a solution in the language of Boolean logic. Let's look at a concrete example first. Suppose you had a Sudoku puzzle in which the upper most left most row and column, respectively, contains 1. The next row down and next column to the right contains 2. Then, continuing along this diagonal, the sequence $3, 7, 4, 8, 9, 5, 6$ occurs so that the entire diagonal beginning from the cell placed at the leftmost and uppermost column and row contains:

$$1, 2, 3, 7, 4, 8, 9, 5, 6.$$

How should we then say that one of the cells is occupied by a digit? Using atomic sentences, and indices for them, let $S_{i,j,k}$ express the claim that the digit k is on the cell (i, j), where i is the row, and j is the column. Hence, for example, the Boolean atomic sentence:

$$S_{1,1,1}$$

expresses the claim or condition that *the number 1 occupies the leftmost column and topmost row.* Likewise, the Boolean atomic sentence:

$$S_{6,6,8}$$

expresses the claim or condition that *the number 8 occupies the 6th column 6th row of the* 9×9 *grid.* Both of these claims are, in the example above, true. Or better, the conditions are satisfied, in the sense that the conjunction:

$$S_{1,1,1} \wedge S_{6,6,8}$$

is satisfied—or made true—by the arrangement in the example above, whereas the Boolean compound sentence:

$$(S_{1,1,1} \wedge S_{6,6,8}) \wedge S_{2,2,3}$$

is not satisfied by the arrangement in the example above. Let's attack the general problem of describing, using Boolean language, a Soduko solution.

We need to satisfy five different conditions. One of them needs to describe what it means for a digit to occupy a cell. Another needs to say that no more than one digit can occupy a cell. Another needs to say that a digit is in a row. Another needs to say that a digit is in a column. The last condition needs to say that each one of the nine digits occurs in each 3×3 subsquare of the 9×9 grid. If we now want to say that one of the nine digits occurs in a cell, we ought to write down a formula that says that either 1 occurs in a cell (i, j), or 2 occurs in a cell (i, j), and so on. That is:

$$S_{i,j,1} \vee S_{i,j,2} \vee S_{i,j,3} \vee S_{i,j,4} \vee S_{i,j,5} \vee S_{i,j,6} \vee S_{i,j,7} \vee S_{i,j,8} \vee S_{i,j,9}.$$

Call that condition (**1**).

Now we need to say that a cell (i, j) cannot contain more than one digit. For example, the compound Boolean sentence:

$$\neg S_{i,j,1} \vee \neg S_{i,j,2} \Leftrightarrow \neg(S_{i,j,1} \wedge S_{i,j,2})$$

expresses the condition that (i, j) cannot contain both 1 and 2. Along with:

$$\neg S_{i,j,1} \vee \neg S_{i,j,3}$$

and $\neg S_{i,j,1} \vee \neg S_{i,j,4}$ and $\neg S_{i,j,1} \vee \neg S_{i,j,5}$ and $\neg S_{i,j,1} \vee \neg S_{i,j,6}$ and $\neg S_{i,j,1} \vee \neg S_{i,j,7}$ and $\neg S_{i,j,1} \vee \neg S_{i,j,8}$ and $\neg S_{i,j,1} \vee \neg S_{i,j,9}$ expresses the condition that if cell (i, j) contains the number 1, then it cannot contain any other number. We need to iterate this process for each number, beginning next at the number 2. Call the set of pairs of disjunctions condition (**2**) and represent it as:

$$\neg S_{i,j,k} \vee \neg S_{i,j,l} \qquad i, j \in \{1, \ldots, 9\} \text{ and } 1 \leq k < l \leq 9.$$

The next two conditions are pretty simple. One of them needs to say that the digit k occurs in one of the rows. For example, the Boolean compound sentence:

$$S_{1,1,k} \vee S_{1,2,k} \vee \ldots \vee S_{1,9,k}$$

expresses the condition that k occurs in row 1, either in column 1, or in column 2, and so on. So let's apply that to each row. Then we'll have a long disjunction:

$$\bigvee_{j \leq 9} S_{i,j,k} \quad i, k \in \{1, \ldots, 9\}$$

that expresses the condition that, as we go through each of the columns, we'll find one of the nine digits on row 1, row 2, and so on. Call that condition (**3**). Our next task is similarly simple. We need to express the condition that k occurs in one of the columns. Just like we did for the rows, then, let's take a long disjunction:

$$\bigvee_{i \leq 9} S_{i,j,k} \quad j, k \in \{1, \ldots, 9\}$$

that expresses the condition that k occurs in one of the columns. Call that condition (**4**). So far, so good. We have managed to come up with four Boolean compound sentences that, together, describe the placement of digits in rows and columns on the entire 9×9 Sudoku board. However,

in order to count as a *solution* to Sudoku, we also need to come up with a condition that describes the fact that each digit from 1 to 9 must occur in each individual cell of the 3×3 subsquares that constitute the board's partition. Hence, we need one more condition that expresses just that.

How should we express the condition that the digit k occurs in one of the nine cells in one of the 3×3 subsquares of the 9×9 grid? Let's take the first subsquare to the right of the uppermost leftmost subsquare. If k occurs in one of these cells, then since on the 9×9 grid, that is columns 4 through 6 and rows 1 through 3, we should express it as a disjunction over these 9 cells. In other words:

$$S_{1,4,k} \vee S_{1,5,k} \vee S_{1,6,k} \vee S_{2,4,k} \vee S_{2,5,k} \vee S_{2,6,k} \vee S_{3,4,k} \vee S_{3,5,k} \vee S_{3,6,k}$$

expresses the condition that we need. A shorter version would use the big disjunction symbol:

$$\bigvee S_{i,j,k} \quad \text{such that } 1 \leq i \leq 3 \text{ and } 4 \leq j \leq 6.$$

Let's try to organize all of the conditions for each subsquare into a single condition. We know that we need to express the condition that the digit k occurs in one of the nine cells of a subsquare. Let's fix the upper left corner of each subsquare and say that the pair (a_s, b_s) describes the upper left corners of the subsquares, where s is the number that occurs in the cell of that subsquare. Then, it is possible to express the condition that the digit k occurs in a 3×3 subsquare by the Boolean compound sentence:

$$\bigvee S_{i,j,k} \quad \text{such that } a_s \leq i \leq a_s + 3 \text{ and } b_s \leq j \leq b_s + 3$$

This expression fixes the range of placing digits 1 through 9 to each subsquare by fixing the rows and columns that constitute one of the 9 different 3×3 subsquares. Call that condition (**5**).

At this point, we have 5 different conditions. One of them describes the condition under which a digit occurs in a cell on the grid. Another describes the condition under which no more than one digit can occur in a cell on the

grid. Another describes the condition under which a digit occurs on a row, another describes the condition under which a digit occurs on a column, and the last describes the condition under which one of the nine digits occurs in a cell on a 3×3 subsquare of the grid. These are the conditions under which, if they are all satisfied together, then we shall have a solution to a Sudoku puzzle. In other words, let's take the conjunction of the five conditions:

$$(1) \wedge (2) \wedge (3) \wedge (4) \wedge (5)$$

and denote that conjunction as **SDK**.

Hence, we're now in a better position to prove the theorem. The next step is to define what we mean by a *satisfaction assignment* for a formula. Look back at the truth tables. The assignment of truth values to atomic sentences under which a conjunction is true is the assignment that sends **T** to both conjuncts. Satisfaction is like truth, in that we say that the conjunction of two sentences is satisfied just in case both conjuncts are satisfied. Let's define a satisfaction assignment accordingly.

Definition: For all formulae α of the Boolean language of sentential logic, v is a satisfaction assignment for α *if and only if* $v(\alpha) = T$

Here, α just acts as a variable for any old sentence of the Boolean language. Thus, for example, α could be $A \vee B$, in which case the satisfaction assignment for α is the assignment v' under which $v'(A) = T$, or $v'(B) = T$, or both. We can now restate the theorem in much clearer language.

Theorem: For all satisfaction assignments v, $v(SDK) = T$ *if and only if* there exists a solution to a Sudoku puzzle.

Note that there are two parts to the proof. For one part we'll assume that we have a satisfaction assignment to **SDK** and show that there must be a solution to the puzzle. For the other part, we'll assume that we have a solution to a Sudoku puzzle and show that there must be a satisfaction assignment to **SDK** such that $v(SDK) = T$. Let's work on the direction from left to right first.

Proof: First note that the left to right direction is a conditional. So we need to assume the antecedent and try to prove the consequent. Hence, assume that we have an arbitrary v' where v' is a satisfaction assignment for **SDK**. In other, more precise, words:

$$v'(SDK) = v'((1) \wedge (2) \wedge (3) \wedge (4) \wedge (5)) = T.$$

Hence, we need to show that given the truth of all five conditions there is a solution, a "correct placement" of digits on the board. Let's be more precise about what "correct placement" means. Let's define a correct placement to be the condition that if $v'(S_{i,j,k}) = T$, then in the square s with cell (i, j), the digit that occurs there is k. That is, define a placement of digits $s_{i,j} \in \{1, \ldots, 9\}$ into cells (i, j) as:

$$s_{i,j} := k \quad \text{iff} \quad v'(S_{i,j,k}) = T.$$

This definition says that the digit k occurs in cell (i, j) within the square s just in case the Boolean sentence $S_{i,j,k}$ is true. Now we need to show that if the five conditions that constitute **SDK** are all satisfied by v', then k will be correctly placed in a cell (i, j) within a square s such that the placement constitutes a solution to a puzzle. That is, we have five different subproofs, corresponding to the five conditions.

1. First we need to show that if $v'((1)) = T$, then there is some k such that $k \in \{1, \ldots, 9\}$. Assume, then, that $v'((1)) = T$. Then at least one of the disjuncts in (1) must be true. Hence, $v'(S_{i,j,k}) = T$ for some k. Since, there are nine disjuncts, it also follows that $k \in \{1, \ldots, 9\}$.
2. Second we need to show that if $v'((2)) = T$, then the k in cell (i, j) is unique. Assume that $v'((2)) = T$. For a contradiction, also assume that v' assigns the value T to two different Boolean sentences. That is, assume, for a contradiction, that $v'(S_{i,j,k}) = T$ and $v'(S_{i,j,l}) = T$ such that $k \neq l$. Then it follows that $v'(\neg S_{i,j,k} \vee \neg S_{i,j,l}) = F$ for that clause in (2). But that contradicts our

assumption that $v'((2)) = T$ for all of the clauses in (2).

3. Third we need to show that if $v'((3)) = T$, then the ith row contains k. Assume that $v'((3)) = T$. Then, it follows that for each k the disjunction in (3) is true. Hence, $v'(S_{i,j,k}) = T$ for some j from 1 to 9. Hence, $s_{i,j} = k$ for each k at the ith row.
4. Fourth we need to show that if $v'((4)) = T$, then the jth column contains k. Assume that $v'((4)) = T$. Then, it follows that for each k the disjunction is true. Hence, $v'(S_{i,j,k}) = T$ for some i from 1 to 9. Hence, $s_{i,j} = k$ for each k at the jth column.
5. Fifth we need to show that if $v'((5)) = T$, then the square fixed by (a_s, b_s) contains each digit. Assume that $v'((5)) = T$. Then, it follows that for each k the disjunction in (5) is true. Hence $v'(S_{i,j,k}) = T$ for some i and some j such that:

$$a_s \leq i \leq a_s + 3 \text{ and } b_s \leq j \leq b_s + 3$$

Hence, $s_{i,j} = k$ for each subsquare fixed by the uppermost leftmost corner at (a_s, b_s).

We have verified that, if each condition from (1) through (5) is true, then the digit k that occurs on a row i in a column j in a subsquare will be unique, and that each digit occurs in a subsquare. Of course, that's just what it *means* to have a solution to a Sudoku puzzle. Hence, one half of our proof is now complete. Q.E.D.

We now have to work on the right to left direction. That is, now we'll assume that we have a correct placement of digits on a given Sudoku board, and show that there is a satisfaction assignment v' for **SDK**.

Proof: Again we need to define a satisfaction assignment to Boolean sentences $S_{i,j,k}$. Let's mimic the definition above. Define an assignment v' such that:

$$v'(S_{i,j,k}) = T \quad \text{iff} \quad s_{i,j} = k.$$

Now we need to check that v' is a satisfaction assignment for all of the conjuncts in **SDK**. Let's start with the first conjunct.

1. First we assume that $s_{i,j}$ is a digit k. It follows that $v'(S_{i,j,k}) = T$ for some k and hence that $v'((1)) = T$, since at least one of its disjuncts is true.
2. Second assume that k is a unique digit. If so, then if $k \neq l$, then either $s_{i,j} \neq k$ or $s_{i,j} \neq l$. It follows that $v'(\neg S_{i,j,k} \vee \neg S_{i,j,l}) = T$. Hence, $v'((2)) = T$.
3. Third assume that each row i for some column j contains digit k. Then, it follows that $s_{i,j} = k$. Hence, $v'(S_{i,j,k}) = T$ for each row i and each k for some column j. Hence, $v'((3)) = T$.
4. Fourth assume that each column j for some row i contains digit k. Then, it follows that $s_{i,j} = k$. Hence, $v'(S_{i,j,k}) = T$ for each column j and each digit k for some row i. Hence, $v'((4)) = T$.
5. Fifth, assume that each subsquare fixed by (a_s, b_s) contains contains each digit k. Then for some cell (i, j) in the subsquare, it follows that $s_{i,j} = k$ such that:

$$a_s \leq i \leq a_s + 3 \text{ and } b_s \leq j \leq b_s + 3.$$

Hence, $v'(S_{i,j,k}) = T$ for some cell (i, j) in the subsquare. Hence, $v'((5)) = T$, since at least one of the disjuncts is true.

We have verified that, if we have a solution to a Sudoku puzzle, that is that if one of the digits 1 through 9 occurs in the cell (i, j) on the 9×9 grid, and no more than one digit occurs in that cell, and that if each digit occurs in a row, and each digit occurs in a column, and that each digit occurs in each cell of each subsquare, then each condition (1) through (5) is true under the satisfaction assignment. Hence, the entire conjunction **SDK** is satisfied by the assignment. Q.E.D.

Our proof is now complete.

What we'd like to do now is verify that the reduction of the combinatorial problem of determining the number of solutions to Sudoku puzzles can be reduced to a much much easier problem: namely, the problem of counting the number of different clauses in **SDK**. What we want to show is that if

n is the number of clauses in **SDK** and m is the cardinality of the set of correct solutions to Sudoku puzzles, then $n < m$. From the computational point of view, this could be helpful, since if it takes less time to count the number of clauses in **SDK** than it does to count the number of correct Sudoku solutions, and since the theorem asserts that **SDK** is a description of the set of solutions, then we can simply replace the job of counting the number of correct solutions to Sudoku by the much easier task of counting the number of clauses in **SDK**. Let's start counting, beginning with the number of clauses in (1), and continuing systematically. Note that a true fact is that the number of correct solutions to Sudoku puzzles is:

$$6.671 \times 10^{21}.$$

1. Condition (1) contains 81 different clauses, one for each cell.

2. Condition (2) contains 81×36 different clauses, since (2) has a clause for all 81 cells (i, j). But each clause in (2) also represents each pair (k, l) where $k < l$ and $1 \leq k < l \leq 9$. That is, there are $\frac{9 \times 8}{2} = 9 \times 4 = 36$ possibilities for each pair (k, l). Hence the total is:

$$81 \times 36 = 2,916.$$

3. Condition (3) contains a clause for each i and each j. Hence, the total number of clauses is:

$$9 \times 9 = 81.$$

4. Condition (4) contains again a clause for each i and each j. Hence, the total number of clauses is:

$$9 \times 9 = 81.$$

5. Condition (5) contains again a clause for each i and each j. Hence, the total number of clauses is:

$$9 \times 9 = 81.$$

Hence the sum total of clauses in **SDK** is:

$$81 + 2916 + 81 + 81 + 81 = 3,240.$$

From a computational point of view, since the number of clauses in **SDK** is far less than the number of solutions to Sudoku puzzles, it is much easier to find a solution by looking through the set of clauses in **SDK** than it is to sift through the number of solutions. This is an illustration of how reducing a combinatorial problem (like Sudoku solutions) to a problem involving satisfaction assignments to Boolean sentences yields computationally more tractable solutions. In other words, it's an illustration of how, despite the fact that it *seems* difficult, logic actually makes things easier.