SAT Solving Sudoku

Aresh Pourkavoos

March 28, 2022

Sudoku is a puzzle game which consists of a 9×9 square grid, subdivided into 3×3 blocks and partially filled with the digits 1-9. The goal is to fill in the rest of the digits such that every row, column and block contains each digit exactly once. Sudoku is a well-studied problem, and there are many algorithms out there to solve it, from pencil-and-paper tricks used mostly by human players to trial-and-error methods used by computers, such as backtracking. However, I will focus on one particular approach: SAT solving.

SAT is short for "satisfiability," and a SAT solver is a program that checks whether a given statement is satisfiable. In this case, a statement is a logical proposition such as "P and Q", and a statement is satisfiable if there is some way to assign each variable (P and Q, in this case) to either true or false such that the entire statement is true. "P and Q," for example, is satisfiable: it has exactly one solution, where P and Q are both true. "P and not P," on the other hand, is not satisfiable, since for both possible values of P, the statement is false. A brute-force SAT solver would simply check every possible combination, of which there are 2^n, where n is the number of variables. However, there are much more optimized algorithms in use, making SAT solvers practical for huge formulas with thousands of variables or more.

However, it's not immediately obvious how Sudoku could be translated into a SAT problem: after all, each cell is filled with one of 9 possible digits, not one of 2 truth values. We can represent a choice of 9 by having 9 variables for each cell, one for each possible digit. But this creates another problem: out of the 512 possible assignments, only 9 are valid. We need to constrain these variables (in typical SAT solver fashion) to say that exactly one variable must be true. There are two parts to this problem: saying that at least one is true and saying that at most one is true. The former can be expressed as an or statment containing the given variables:

$$p_1 \vee p_2 \vee \ldots \vee p_n$$

The latter looks at all pairs of variables, saying that they can't both be true:

$$\neg(p_1 \wedge p_2) \wedge \ldots \wedge \neg(p_1 \wedge p_n) \wedge \neg(p_2 \wedge p_3) \wedge \ldots \wedge \neg(p_2 \wedge p_n) \wedge \ldots \wedge \neg(p_{n-1} \wedge p_n)$$

For n variables, there are $\binom{n}{2} = \frac{n(n-1)}{2}$ pairs. For a given Sudoku cell, where there are 9 variables, it takes $\binom{9}{2}$ of them to say that at most one variable may be true, and the or clause containing all 9 variables makes 37. Since there are 81 cells, it takes $37 \times 81 = 2997$ formulas just to say that every cell is filled with exactly one digit. To express the rest of the puzzle, similar constraints must be applied to other sets of 9 variables:

- Every row of cells contains every digit exactly once, i.e. for every row and digit, exactly 1 of the 9 variables that represent filling a cell within the given row with a given digit must be true.
- Every column contains every digit exactly once, i.e. for every column and digit, exactly 1 of the 9 variables that represent filling a cell within the given column with a given digit must be true.
- Every 3×3 block contains every digit exactly once, i.e. for every block and digit, exactly 1 of the 9 variables that represent filling a cell within the given block with a given digit must be true.

The following table shows a way to formulate the statement that at most one of 8 variables $a_0, \ldots, a_7$ is true, using a binary divide-and-conquer method.

$$
\begin{array}{r}
a_0, a_1 \text{ free}| \\
\underline{\neg a_0 \vee \neg a_1|} \\
a_{0-1} := a_0 \vee a_1 \\
\neg a_{0-1} \vee a_0 \vee a_1 \\
\neg a_0 \vee a_{0-1} \\
\neg a_1 \vee a_{0-1}
\end{array}
\qquad
\begin{array}{r}
a_2, a_3 \text{ free}| \\
\neg a_2 \vee \neg a_3| \\
a_{2-3} := a_2 \vee a_3| \\
\neg a_{2-3} \vee a_2 \vee a_3| \\
\neg a_2 \vee a_{2-3}| \\
\neg a_3 \vee a_{2-3}| \\
\underline{\neg a_{0-1} \vee \neg a_{2-3}|} \\
a_{0-3} := a_{0-1} \vee a_{2-3} \\
\neg a_{0-3} \vee a_{0-1} \vee a_{2-3} \\
\neg a_{0-1} \vee a_{0-3} \\
\neg a_{2-3} \vee a_{0-3}
\end{array}
\qquad
\begin{array}{r}
a_4, a_5 \text{ free} \\
\neg a_4 \vee \neg a_5 \\
a_{4-5} := a_4 \vee a_5 \\
\neg a_{4-5} \vee a_4 \vee a_5 \\
\neg a_4 \vee a_{4-5} \\
\neg a_5 \vee a_{4-5}
\end{array}
\qquad
\begin{array}{r}
a_6, a_7 \text{ free}| \\
\neg a_6 \vee \neg a_7| \\
a_{6-7} := a_6 \vee a_7| \\
\neg a_{6-7} \vee a_6 \vee a_7| \\
\neg a_6 \vee a_{6-7}| \\
\neg a_7 \vee a_{6-7}| \\
\neg a_{4-5} \vee \neg a_{6-7}| \\
a_{4-7} := a_{4-5} \vee a_{6-7}| \\
\neg a_{4-7} \vee a_{4-5} \vee a_{6-7}| \\
\neg a_{4-5} \vee a_{4-7}| \\
\neg a_{6-7} \vee a_{4-7}| \\
\underline{\neg a_{0-3} \vee \neg a_{4-7}|}
\end{array}
$$

i	Free vars	Aux vars	Formulas
1	2	0	1
2	4	2	9
3	8	6	25

At level i of the tree, there are $n = 2^i$ free variables, $n - 2$ auxiliary variables (for a total of $2n - 2$ variables), and $4n - 7$ formulas. In contrast, the naive method with no auxiliary variables uses $1 + (n^2 - n)/2$ formulas.

$$
\begin{array}{r}
a_0, a_1, a_2 \text{ free}| \\
\neg a_0 \vee \neg a_1| \\
\neg a_0 \vee \neg a_2| \\
\underline{\neg a_1 \vee \neg a_2|} \\
a_{0-2} := a_0 \vee a_1 \vee a_2 \\
\neg a_{0-2} \vee a_0 \vee a_1 \vee a_2 \\
\neg a_0 \vee a_{0-2} \\
\neg a_1 \vee a_{0-2} \\
\neg a_2 \vee a_{0-2}
\end{array}
\qquad
\begin{array}{r}
a_3, a_4, a_5 \text{ free} \\
\neg a_3 \vee \neg a_4 \\
\neg a_3 \vee \neg a_5 \\
\neg a_4 \vee \neg a_5 \\
a_{3-5} := a_3 \vee a_4 \vee a_5 \\
\neg a_{3-5} \vee a_3 \vee a_4 \vee a_5 \\
\neg a_3 \vee a_{3-5} \\
\neg a_4 \vee a_{3-5} \\
\neg a_5 \vee a_{3-5}
\end{array}
\qquad
\begin{array}{r}
a_6, a_7, a_8 \text{ free}| \\
\neg a_6 \vee \neg a_7| \\
\neg a_6 \vee \neg a_8| \\
\neg a_7 \vee \neg a_8| \\
a_{6-8} := a_6 \vee a_7 \vee a_8| \\
\neg a_{6-8} \vee a_6 \vee a_7 \vee a_8| \\
\neg a_6 \vee a_{6-8}| \\
\neg a_7 \vee a_{6-8}| \\
\neg a_8 \vee a_{6-8}| \\
\neg a_{0-2} \vee \neg a_{3-5}| \\
\neg a_{0-2} \vee \neg a_{6-8}| \\
\underline{\neg a_{3-5} \vee \neg a_{6-8}|}
\end{array}
$$

i	Free vars	Aux vars	Formulas
1	3	0	3
2	9	3	24

At level i of the tree, there are $n = 3^i$ free variables, $(n-3)/2$ auxiliary variables (for a total of $(3n-3)/2$ variables), and $(7n - 15)/2$ formulas.