3 Encoding constraint problems into satisfiability problems

In order to illustrate the concepts that we have seen so far, we are now going to discuss how some well-known constraint problems can be encoded in propositional logic, and how we can reason about them using our machinery.

Sudoku

The first problem that we encode into propositional logic is the well-known Sudoku puzzle:

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

For each $i, j, k \in \{1, \ldots, 9\}$ we have a proposition $x_{i,j,k}$ expressing that *grid position i, j contains number k*. Now build a formula F as the conjunction of the following *constraints*: 2.16

- Each number appears in each row and in each column:

$$F_1 := \bigwedge_{i=1}^{9} \bigwedge_{k=1}^{9} \bigvee_{j=1}^{9} x_{i,j,k} \qquad F_2 := \bigwedge_{j=1}^{9} \bigwedge_{k=1}^{9} \bigvee_{i=1}^{9} x_{i,j,k}$$

- Each number appears in each 3×3 block:

$$F_3 := \bigwedge_{k=1}^{9} \bigwedge_{u=0}^{2} \bigwedge_{v=0}^{2} \bigvee_{i=1}^{3} \bigvee_{j=1}^{3} x_{3u+i,3v+j,k}$$

- No square contains two numbers:

$$F_4 := \bigwedge_{i=1}^{9} \bigwedge_{j=1}^{9} \bigwedge_{1 \leq k < k' \leq 9} \neg(x_{i,j,k} \wedge x_{i,j,k'}) \,.$$

- Certain numbers appear in certain positions: we assert 2.17

$$F_5 := x_{2,1,2} \wedge x_{1,2,8} \wedge x_{2,3,3} \wedge \ldots \wedge x_{8,9,6} \,.$$

The formula F thus obtained is satisfiable if and only if the given Sudoku instance has a solution. Some facts about Sudokus have not explicitly been included in F. For instance, the property that no number appears twice in the same row, i.e., 2.18

$$F_6 := \bigwedge_{i=1}^{9} \bigwedge_{k=1}^{9} \bigwedge_{1 \leq j < j' < 9} \neg(x_{i,j,k} \wedge x_{i,j',k})$$

has not explicitly been stated. However, it can be verified that $\models F \to F_6$. Though redundant, explicitly adding F_6 to F may help a satisfiability solver when searching for a satisfying assignment. Note that the number of variables $x_{i,j,k}$ is $9^3 = 729$. Thus a truth table for the corresponding formula would have $2^{729} > 10^{200}$ lines! Nevertheless a modern SAT-solver can find a satisfying assignment in milliseconds. 2.19