A Formal Reduction of Sudoku Puzzle into SAT

M. Fareed Arif

TSSG
Security Research Group,
ArcLabs Research and Innovation Centre,
Waterford Institute of Technology (WIT),
Carriganore Campus, Co. Waterford, Ireland.

May 25, 2012

Outline

1. Demonstration: Sudoku Puzzle Solver
2. Preliminaries: SAT
3. Formal Reduction: Encoding Sudoku in SAT
4. Applications of SAT
    - Search & Planning
    - Model Checking
    - Zero Knowledge Proof
    - Complexity Theory
5. Appendix

Sudoku Puzzle

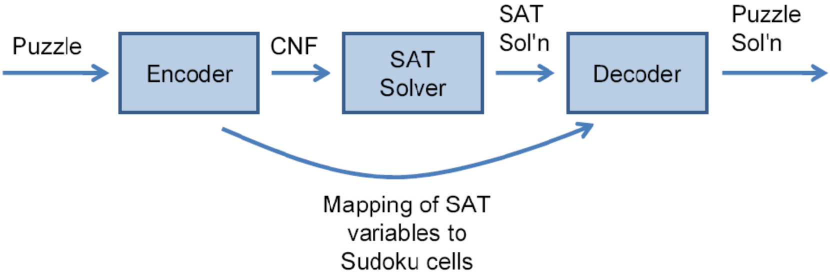
		4				5	3	
6			1	3				9
	7					2		
				8		4	7	
			3		4			
	5	9		6				
		6					9	
9				7	1			5
	2	7				1		

Sudoku Puzzle

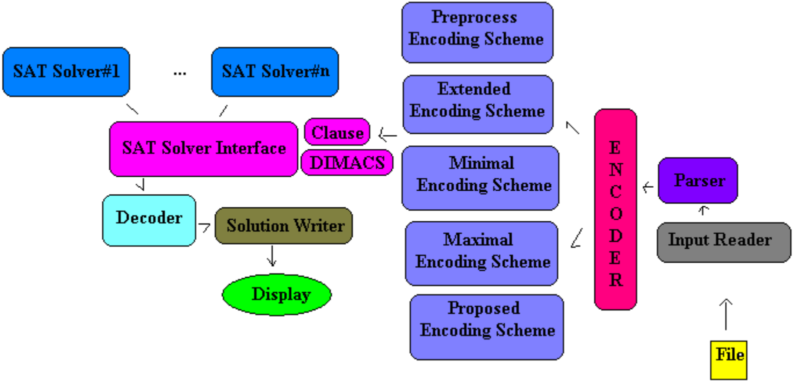
		4				5	3	
6			1	3				9
	7					2		
				8		4	7	
			3		4			
	5	9		6				
		6					9	
9				7	1			5
	2	7				1		

6.10^{21} possible grids for a simple looking $9 * 9$ puzzle instance.

Abstract Model



Sudoku Solver Architecture



Preliminaries: Propositional Logic

Vocabulary

An alphabet of propositional logic consists of

- a (countable) infinite set $\mathcal{R} = \{p_0, p_1, p_2, \ldots\}$ of propositional variables,
- the set logical connectives: $\circ \in \{\neg/1, \wedge/2, \vee/2, \rightarrow /2, \leftrightarrow /2$
- the special characters "(" and ")".

Syntax

An atomic formula, atom, is a propositional variable.
The set of propositional formulas is the smaller set $\mathcal{L}(\mathcal{R})$ of strings over an alphabet of propositional logic with the following properties:

- if F is an atomic formula, then $F \in \mathcal{L}(\mathcal{R})$.
- if $F \in \mathcal{L}(\mathcal{R})$, then $\neg F \in \mathcal{L}(\mathcal{R})$.
- if $\circ/2$ is a binary connective, $F, G\mathcal{L}(\mathcal{R})$, then $(F \circ G) \in \mathcal{L}(\mathcal{R})$.

Truth Table Semantics

- The set of truth values $\mathcal{W}$ is the set $\{\top, \bot\}$.
- We consider the following functions on $\mathcal{W}$:
  - Negation $\neg^*/1$.
  - Conjunction $\wedge^*/2$.
  - Disjunction $\vee^*/2$.
  - Implication $\rightarrow^*/2$.
  - Equivalence $\leftrightarrow^*/2$.

		$\neg^*$	$\wedge^*$	$\vee^*$	$\rightarrow^*$	$\leftrightarrow^*$
$\top$	$\top$	$\bot$	$\top$	$\top$	$\top$	$\top$
$\top$	$\bot$	$\bot$	$\bot$	$\top$	$\bot$	$\bot$
$\bot$	$\top$	$\top$	$\bot$	$\top$	$\top$	$\bot$
$\bot$	$\bot$	$\top$	$\bot$	$\bot$	$\top$	$\top$

Model based Semantics

An interpretation $\mathcal{I} = (\mathcal{W}, .^I)$ is a mapping $.^I : \mathcal{L}(\mathcal{R}) \rightarrow \mathcal{W}$ such that:

SAT

A propositional satisfiability problem (SAT), consist of a formula $\phi \in \mathcal{L}(\mathcal{R})$, and is the problem to decide whether ϕ is satisfiable.

Model

An interpretation $\mathcal{I} = (\mathcal{W}, .^I)$ is called a model for propositional formula ϕ , $\mathcal{I} \models \phi$ if $[\phi]^I = \top$ (i.e., $\mathcal{I}$ satisfies ϕ). ϕ is unsatisfiable if it has no models.

Propostional Satisfiability Problems

SAT is a combinatorial decision problem.

- Decision variant yes/no answer.
- Search variant find a model if ϕ is satisfiable.

Example

- Let $\{p_1, p_2, p_3, p_4, p_5\} \subseteq \mathcal{R}$.

$$\begin{aligned}\phi = & (\neg p_1 \vee p_2) \wedge (\neg p_2 \vee p_1) \\ & \wedge (\neg p_1 \vee \neg p_2 \vee \neg p_3) \wedge (p_1 \vee p_2) \\ & \wedge (\neg p_4 \vee p_3) \wedge (\neg p_5 \vee p_3)\end{aligned}$$

Propostional Satisfiability Problems

SAT is a combinatorial decision problem.

- Decision variant yes/no answer.
- Search variant find a model if ϕ is satisfiable.

Example

- Let $\{p_1, p_2, p_3, p_4, p_5\} \subseteq \mathcal{R}$.

$$\begin{aligned}\phi = &(\neg p_1 \vee p_2) \wedge (\neg p_2 \vee p_1) \\ &\wedge (\neg p_1 \vee \neg p_2 \vee \neg p_3) \wedge (p_1 \vee p_2) \\ &\wedge (\neg p_4 \vee p_3) \wedge (\neg p_5 \vee p_3)\end{aligned}$$

- $\{p_1, p_2\}$ is a model of ϕ.

Hence, ϕ is satisfiable.

Formal Reduction

Idea:

Sudoku puzzle $\mathcal{S}$ is formulated as a CNF formula ϕ such that is

$$\phi \texttt{ is satisfiable iff } \mathcal{S} \texttt{ has a solution.}$$

Sudoku Puzzle $\mathcal{S}$:

A Sudoku puzzle $\mathcal{S}$ is represented by a $\mathbb{N} * \mathbb{N}$ grid, which comprises of an $\sqrt{\mathbb{N}} * \sqrt{\mathbb{N}}$ sub-grids (also called boxes). Some of the entries in the grid are filled with numbers from 1 to $\mathbb{N}$, whereas other entries are left blank.

Encoding Scheme ($\mathcal{S} \implies \phi$):

A SAT problem is represented as a propositional formula Φ where each variable P_i is assigned 0 ($\mathbb{F}$) or 1 ($\mathbb{T}$) where i $\in (1, \cdots, n)$ In Sudoku each tuple (r, c, v) denotes a variable which is true iff the cell in row r and column c is assigned a number v; $[r, c] = v$. The resulting set of formulas turn out to be $V = \{(r, c, v) \mid 1 \leq r, c, v \leq \mathrm{n}\}$.

Encoding Scheme ($\mathcal{S} \implies \phi$):

- There is at exactly one number in each cell
  $\phi_{cell.ex} := \phi_{cell.def} \wedge \phi_{cell.uniq}$
  There is at least one number for each cell
  $\phi_{cell.def} := \bigwedge_{r=1}^{n} \bigwedge_{c=1}^{n} \bigvee_{v=1}^{n} (r, c, v)$
  Each number appears at most one in each cell
  $\phi_{cell.uniq} := \bigwedge_{r=1}^{n} \bigwedge_{c=1}^{n} \bigwedge_{v_i=1}^{(n-1)} \bigwedge_{v_j=v_i+1}^{n} \neg(r, c, v_i) \vee \neg(r, c, v_j)$
- There is at exactly one number in each row
- There is at exactly one number in each column
- There is at exactly one number in each block

Encodings (ϕ):

$$\Phi_{extended} := \phi_{cell.ex} \wedge \phi_{row.ex} \wedge \phi_{col.ex} \wedge \phi_{blk.ex} \wedge \phi_{assign}.$$

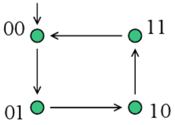
$$\Phi_{efficient} := \phi_{cell.ex} \wedge \phi_{row.uniq} \wedge \phi_{col.uniq} \wedge \phi_{blk.uniq} \wedge \phi_{assign}.$$

$$\Phi_{minimal} := \phi_{cell.def} \wedge \phi_{row.uniq} \wedge \phi_{col.uniq} \wedge \phi_{blk.uniq} \wedge \phi_{assign}.$$

Applications

- Search & Planning
- Model Checking
- Zero Knowledge Proof $\mathcal{S} \overset{\Pi}{\leftarrow} V$
- Complexity Theory ($\mathrm{P} = \mathrm{NP}$ or $\mathrm{P} \neq \mathrm{NP}$)

Model Checking



Initial state: $I : \neg l \wedge \neg r$

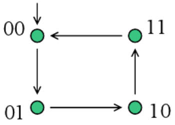
Transition: $R: \begin{pmatrix} l' = (l \neq r) \wedge \\ r' = \neg r \end{pmatrix}$

Safety property: $\mathbf{AG}\,(\neg l \vee \neg r)$

$$\Omega(2) : (\neg l_0 \wedge \neg r_0) \wedge \begin{pmatrix} l_1 = (l_0 \neq r_0) \wedge r_1 = \neg r_0 \wedge \\ l_2 = (l_1 \neq r_1) \wedge r_2 = \neg r_1 \end{pmatrix} \wedge \begin{pmatrix} (l_0 \wedge r_0) \vee \\ (l_1 \wedge r_1) \vee \\ (l_2 \wedge r_2) \end{pmatrix}$$

$\Omega(2)$ is unsatisfiable. $\Omega(3)$ is satisfiable.

Model Checking



Initial state: $I : \neg l \wedge \neg r$

Transition: $R : \begin{pmatrix} l' = (l \neq r) \wedge \\ r' = \neg r \end{pmatrix}$

Safety property: $\mathbf{AG}\ (\neg l \vee \neg r)$

$$\Omega(2) : (\neg l_0 \wedge \neg r_0) \wedge \begin{pmatrix} l_1 = (l_0 \neq r_0) \wedge r_1 = \neg r_0 \wedge \\ l_2 = (l_1 \neq r_1) \wedge r_2 = \neg r_1 \end{pmatrix} \wedge \begin{pmatrix} (l_0 \wedge r_0) \vee \\ (l_1 \wedge r_1) \vee \\ (l_2 \wedge r_2) \end{pmatrix}$$

$\Omega(2)$ is unsatisfiable. $\Omega(3)$ is satisfiable.

Satisfying assignment gives the counter example to the safety property.

$$\mathcal{M} = \{r_1, l_2, l_3, r_3\}$$
$$\mathcal{M} = \{(\neg l_0, \neg r_0), (\neg l_1, r_1), (l_2, \neg r_2), (l_3, r_3)\}$$

For Further Reading

Stephen A. Cook. 1970
The complexity of theorem-proving procedures..

Armin Biere, Marjin Heule, Hans van Marren and Toby Walsh. 2009
Handbook of Satisfiability.

Q & A