SAT Encodings for Sudoku

Bug Catching in 2006 Fall

Sep. 26, 2006

Gi-Hwon Kwon
Agenda

• Introduction

• Background and Previous Encodings

• Optimized Encoding

• Experimental Results

• Conclusions
What is Sudoku?

Given a problem, the objective is to find a satisfying assignment w.r.t. Sudoku rules.

Sudoku rules

- There is a number in each **cell**.
- A number appears once in each **row**.
- A number appears once in each **column**.
- A number appears once in each **block**.
Sudoku as SAT Problem

1. Sudoku is encoded into CNF form.
2. The CNF is then passed to the SAT solver.
3. The SAT solver checks if there is a model.
4. If yes, the solution is decoded.
5. If no, there is no solution found.

The symbol table links the original Sudoku to the CNF, and the model is used to check for solutions.
Previous Encodings

Sudoku → Encoder → CNF → SAT Solver → SAT? → Decoder

- Minimal encoding [Lynce & Ouaknine, 2006]
- Extended encoding [Lynce & Ouaknine, 2006]
- Efficient encoding [Weber, 2005]
## Analysis of Previous Encodings

<table>
<thead>
<tr>
<th>Encoding</th>
<th>Number of Variables</th>
<th>Number of Clauses</th>
</tr>
</thead>
<tbody>
<tr>
<td>Minimal</td>
<td>$N^3$</td>
<td>$N * N + \left( N * N * \left( \frac{N * (N - 1)}{2} \right) \right) * 3 + k$</td>
</tr>
<tr>
<td>Efficient</td>
<td>$N^3$</td>
<td>$N * N + \left( N * N * \left( \frac{N * (N - 1)}{2} \right) \right) * 4 + k$</td>
</tr>
<tr>
<td>Extended</td>
<td>$N^3$</td>
<td>$\left( N * N + N * N * \left( \frac{N * (N - 1)}{2} \right) \right) * 4 + k$</td>
</tr>
</tbody>
</table>
Exponential Growth in Clauses

<table>
<thead>
<tr>
<th>size</th>
<th>minimal</th>
<th>efficient</th>
<th>extended</th>
</tr>
</thead>
<tbody>
<tr>
<td>9x9</td>
<td>8829</td>
<td>11745</td>
<td>11988</td>
</tr>
<tr>
<td>16x16</td>
<td>92416</td>
<td>123136</td>
<td>123904</td>
</tr>
<tr>
<td>25x25</td>
<td>563125</td>
<td>750625</td>
<td>752500</td>
</tr>
<tr>
<td>36x36</td>
<td>2450736</td>
<td>3267216</td>
<td>3271104</td>
</tr>
<tr>
<td>49x49</td>
<td>8473129</td>
<td>11296705</td>
<td>11303908</td>
</tr>
<tr>
<td>64x64</td>
<td>24776704</td>
<td>33034240</td>
<td>33046528</td>
</tr>
<tr>
<td>81x81</td>
<td>63779481</td>
<td>85037121</td>
<td>85056804</td>
</tr>
</tbody>
</table>
## Experimental Results

<table>
<thead>
<tr>
<th>Size</th>
<th>Level</th>
<th>Variables</th>
<th>Clauses</th>
<th>Time</th>
<th>Variables</th>
<th>Clauses</th>
<th>Time</th>
<th>Variables</th>
<th>Clauses</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>9x9</td>
<td>Easy</td>
<td>729</td>
<td>8854</td>
<td>0.00</td>
<td>729</td>
<td>11770</td>
<td>0.00</td>
<td>729</td>
<td>12013</td>
<td>0.00</td>
</tr>
<tr>
<td>9x9</td>
<td>Hard</td>
<td>729</td>
<td>8859</td>
<td>0.00</td>
<td>729</td>
<td>11775</td>
<td>0.00</td>
<td>729</td>
<td>12018</td>
<td>0.00</td>
</tr>
<tr>
<td>16x16</td>
<td>Easy</td>
<td>4096</td>
<td>92520</td>
<td>0.10</td>
<td>4096</td>
<td>123240</td>
<td>0.09</td>
<td>4096</td>
<td>124008</td>
<td>0.01</td>
</tr>
<tr>
<td>16x16</td>
<td>Hard</td>
<td>4096</td>
<td>92514</td>
<td>0.46</td>
<td>4096</td>
<td>123234</td>
<td>0.21</td>
<td>4096</td>
<td>124002</td>
<td>0.01</td>
</tr>
<tr>
<td>25x25</td>
<td>Easy</td>
<td>15625</td>
<td>563417</td>
<td>9.07</td>
<td>15625</td>
<td>750917</td>
<td>17.48</td>
<td>15625</td>
<td>752792</td>
<td>0.07</td>
</tr>
<tr>
<td>25x25</td>
<td>Hard</td>
<td>15625</td>
<td>563403</td>
<td>Time</td>
<td>15625</td>
<td>750903</td>
<td>Time</td>
<td>15625</td>
<td>752778</td>
<td>0.21</td>
</tr>
<tr>
<td>36x36</td>
<td>Easy</td>
<td>46656</td>
<td>2451380</td>
<td>Time</td>
<td>46656</td>
<td>3267860</td>
<td>Time</td>
<td>46656</td>
<td>3271748</td>
<td>0.50</td>
</tr>
<tr>
<td>36x36</td>
<td>Hard</td>
<td>46656</td>
<td>2451400</td>
<td>Time</td>
<td>46656</td>
<td>3267880</td>
<td>Time</td>
<td>46656</td>
<td>3271768</td>
<td>0.67</td>
</tr>
<tr>
<td>49x49</td>
<td>Easy</td>
<td>117649</td>
<td>8474410</td>
<td>Time</td>
<td>117649</td>
<td>11297986</td>
<td>Time</td>
<td>117649</td>
<td>11305189</td>
<td>1.47</td>
</tr>
<tr>
<td>64x64</td>
<td>Easy</td>
<td>262144</td>
<td>24779088</td>
<td>Stack</td>
<td>262144</td>
<td>33036624</td>
<td>Stack</td>
<td>262144</td>
<td>33048912</td>
<td>Stack</td>
</tr>
<tr>
<td>81x81</td>
<td>Easy</td>
<td>531441</td>
<td>63783464</td>
<td>Stack</td>
<td>531441</td>
<td>85041104</td>
<td>Stack</td>
<td>531441</td>
<td>85060787</td>
<td>Stack</td>
</tr>
</tbody>
</table>
## Experimental Results

<table>
<thead>
<tr>
<th>size</th>
<th>level</th>
<th>minimal encoding</th>
<th>efficient encoding</th>
<th>extended encoding</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>vars</td>
<td>clauses</td>
<td>time</td>
</tr>
<tr>
<td>9x9</td>
<td>easy</td>
<td>729</td>
<td>8854</td>
<td>0.00</td>
</tr>
<tr>
<td>9x9</td>
<td>hard</td>
<td>729</td>
<td>8859</td>
<td>0.00</td>
</tr>
<tr>
<td>16x16</td>
<td>easy</td>
<td>4096</td>
<td>92520</td>
<td>0.10</td>
</tr>
<tr>
<td>16x16</td>
<td>hard</td>
<td>4096</td>
<td>92514</td>
<td>0.46</td>
</tr>
<tr>
<td>25x25</td>
<td>easy</td>
<td>15625</td>
<td>563417</td>
<td>9.07</td>
</tr>
<tr>
<td>25x25</td>
<td>hard</td>
<td>15625</td>
<td>563403</td>
<td>time</td>
</tr>
<tr>
<td>36x36</td>
<td>easy</td>
<td>46656</td>
<td>2451380</td>
<td></td>
</tr>
<tr>
<td>36x36</td>
<td>hard</td>
<td>46656</td>
<td>2451400</td>
<td></td>
</tr>
<tr>
<td>49x49</td>
<td>easy</td>
<td>117649</td>
<td>8474410</td>
<td></td>
</tr>
<tr>
<td>64x64</td>
<td>easy</td>
<td>262144</td>
<td>stack</td>
<td>6624</td>
</tr>
<tr>
<td>81x81</td>
<td>easy</td>
<td>531441</td>
<td>stack</td>
<td>65788456</td>
</tr>
</tbody>
</table>

**Solution found**

**No solution found**
Motivations

• Sudoku was regarded as SAT problem
  ▪ Lynce & Ouaknine, Sudoku as a SAT Problem, Jan. 2006.
  ➔ Extended encoding shows the best performance in our experiments

• Problems in previous works
  ▪ Too many clauses are generated (e.g. 85,056,804 clauses)
  ▪ Thus, large size puzzles are not solved
  ➔ The extended encoding must be optimized for large size puzzles

• Our objectives
  ▪ To propose an optimization for the extended encoding
Agenda

• Introduction

• Background and Previous Encodings
  • Optimized Encoding

• Experimental Results

• Conclusions
Encoding

• Knowledge compilation into a **target language**

  problem knowledge $\rightarrow$ CNF

• Knowledge about Sudoku

- A number appears once in each **cell**
- A number appears once in each **row**
- A number appears once in each **col**
- A number appears once in each **block**
- A pre-assigned number

rules $\rightarrow$ CNF
facts $\rightarrow$ CNF
Glance at CNF

- **CNF** represented by **boolean variables**
  \[ x_1, x_2, \ldots, x_n \]

- **Literal** is a variable or its negation
  \[ x_1 \]
  \[ x_1 = 1 \]

- **Clause** is a disjunction of literals
  \[ \neg x_1 \vee x_2 \vee \neg x_3 \]
  \[ x_1 = 0 \text{ or } x_2 = 1 \text{ or } x_3 = 0 \]

- **CNF** is a conjunction of clauses
  \[ (\neg x_1 \vee x_2) \wedge (x_1 \vee \neg x_3) \]
  \[ (x_1 = 0 \text{ or } x_2 = 1) \text{ and } (x_1 = 1 \text{ or } x_3 = 0) \]
Variables

- Each cell has one number from 1..N
  - \([1,1] = 1\) or \([1,1] = 2\) or … or \([1,1] = N\)
  - Each cell needs N boolean variables to consider all cases

- Total number of variables
  - \(N^3\)

- Boolean variable name as a triple
  - \((r,c,v)\) iff \([r,c] = v\)
  - \(\neg(r,c,v)\) iff \([r,c] \neq v\)
Set Notation

- Indexed generalized disjunction and union operators

\[ \bigcup_{i=1}^{n} l_i = l_1 \lor \ldots \lor l_n \]

\[ \bigcup_{i=1}^{n} C_i = \{ C_1 \} \cup \ldots \cup \{ C_n \} = \{ C_1, \ldots, C_n \} \]

\[ \bigcup_{i=1}^{n} \bigcup_{j=1}^{m} C_{ij} = \{ C_{11}, \ldots, C_{1m}, \ldots, C_{n1}, \ldots, C_{nm} \} \]

- Re-definitions of clause and CNF using the operators

\[ C = \bigcup_{i=1}^{n} l_i = l_1 \lor \ldots \lor l_n \]

\[ \varnothing = \bigcup_{i=1}^{n} C_i = \{ C_1, \ldots, C_n \} \]
Cell Rule $\rightarrow$ CNF

A number appears once in each cell

There is at least one number in each cell  
(definedness)

$$Cell_d = \bigcup_{r=1}^{N} \bigcup_{c=1}^{N} \{ \mathcal{A}_{v=1}^N (r, c, v) \}$$

There is at most one number in each cell  
(uniqueness)
Row Rule → CNF

Each number appears **at least** in each row (definedness)

\[
Row_d = \bigcup_{r=1}^{N} \bigcup_{v=1}^{N} \{ \forall_{c=1}^{N} (r, c, v) \}
\]

Each number appears **at most** in each row (uniqueness)

A number appears once in each row
### Column Rule → CNF

Each number appears **at least** in each column (definedness)

\[
Col_d = \cup_{c=1}^{N} \bigcup_{\nu=1}^{N} \{n_{r=1}^{N} (r, c, \nu)\}
\]

Each number appears **at most** in each column (uniqueness)
Block Rule $\rightarrow$ CNF

A number appears once in each block

Each number appears **at least** in each block (definedness)

Each number appears **at most** in each block (uniqueness)

$\text{Definition: } \forall_{\mathbf{r}, \mathbf{c}} \mathbf{v} \subseteq \{1, \ldots, N\} \forall_{\mathbf{r}, \mathbf{c}} \mathbf{v} \subseteq \{1, \ldots, N\}$
Pre-Assigned Fact $\rightarrow$ CNF

A pre-assigned number

As a constant; the number is never changed

It can be represented as a **unit clause**

$$\text{Assigned} = \bigwedge_{i=1}^{k} \{(r, c, a) \mid \exists 1 \leq a \leq N \cdot [r, c] = a\}$$

where $k$ is a number of pre-assigned numbers
Previous Encodings

**Minimal encoding**  [Lynce & Ouaknine, 2006]

\[ \varphi = \text{Cell}_d \cup \text{Row}_u \cup \text{Col}_u \cup \text{Block}_u \cup \text{Assigned} \]

sufficient to characterize the puzzle

**Extended encoding**  [Lynce & Ouaknine, 2006]

\[ \varphi = \text{Cell}_d \cup \text{Cell}_u \cup \text{Row}_d \cup \text{Row}_u \cup \text{Col}_d \cup \text{Col}_u \cup \text{Block}_d \cup \text{Block}_u \cup \text{Assigned} \]

minimal encoding with redundant clauses

**Efficient encoding**  [Weber, 2005]

\[ \varphi = \text{Cell}_d \cup \text{Cell}_u \cup \text{Row}_u \cup \text{Col}_u \cup \text{Block}_u \cup \text{Assigned} \]

between minimal encoding and extended encoding
## Analysis (Recap)

<table>
<thead>
<tr>
<th>Encoding</th>
<th>Number of Variables</th>
<th>Number of Clauses</th>
</tr>
</thead>
<tbody>
<tr>
<td>Minimal</td>
<td>$N^3$</td>
<td>$N \cdot N + \left( N \cdot N \cdot \left( \frac{N \cdot (N - 1)}{2} \right) \right) \cdot 3 + k$</td>
</tr>
<tr>
<td>Efficient</td>
<td>$N^3$</td>
<td>$N \cdot N + \left( N \cdot N \cdot \left( \frac{N \cdot (N - 1)}{2} \right) \right) \cdot 4 + k$</td>
</tr>
<tr>
<td>Extended</td>
<td>$N^3$</td>
<td>$\left( N \cdot N + N \cdot N \cdot \left( \frac{N \cdot (N - 1)}{2} \right) \right) \cdot 4 + k$</td>
</tr>
</tbody>
</table>
Agenda

• Introduction

• Background and Previous Encodings

• Optimized Encoding

• Experimental Results

• Conclusions
Example

For example, consider the cell [1,1]

- Four cases are considered; thus, four variables are needed
  - (1,1,1), (1,1,2), (1,1,3), (1,1,4)
Variables

• A pre-assigned cell reduces the cases to be considered
  ▪ Because the cell has a fixed number
  ▪ The pre-assigned cell does not need a variable at all
  ▪ It affects other cells located at the same row, or column, or block.

• For example, consider the cell [1,1]
  ▪ The case [1,1]=1 is not allowed since [4,1]=1 are already assigned
  ▪ The case [1,1]=3 is not allowed since [1,4]=3 are already assigned
  ▪ The case [1,1]=4 is not allowed since [1,3]=4 are already assigned
  ▪ Thus, the only case to be considered is [1,1]=2

(1,1,2)
Variables

• Let $V$ be a set of variables

$$V = \bigcup_{r=1}^{N} \bigcup_{c=1}^{N} \bigcup_{v=1}^{N} \{(r, c, v) \mid [r, c] = \text{empty} \land \neg \text{affected} (r, c, v)\}$$

$$\text{affected} (r, c, v) = \text{sameRow} (r, c, v) \lor \text{sameCol} (r, c, v) \lor \text{sameBlock} (r, c, v)$$

$$\text{sameRow} (r, c, v) = \exists_{i:1..N} \cdot i \neq c \Rightarrow [r, i] = v$$

$$\text{sameCol} (r, c, v) = \exists_{i:1..N} \cdot i \neq r \Rightarrow [i, c] = v$$

$$\text{sameBlock} (r, c, v) = \exists_{i: \text{originRow}.. \text{subN}} \cdot \exists_{i: \text{originCol}.. \text{subN}} \cdot (i \neq r \land j \neq c) \Rightarrow [i, j] = v$$
**Example**

\[ V = \{ \begin{array}{c|c|c}
(1,1,2) & (1,2,1) & 4 \\
(1,2,2) & (2,2,1) & 3 \\
(2,1,2) & (2,2,2) & (2,3,1) \\
(2,1,3) & (2,2,4) & (2,3,2) \\
(2,1,4) & & (2,4,1) \\
(3,1,2) & (3,2,2) & (3,3,1) \\
(3,1,3) & (3,2,4) & (3,3,2) \\
(3,1,4) & & (3,3,3) \\
(3,2,1) & & (3,4,1) \\
(3,2,2) & & (3,4,2) \\
(3,2,4) & & (3,4,4) \\
1 & 3 & (4,3,2) \\
& & (4,4,2) \\
& & (4,4,4)
\end{array} \} \]

these parts are excluded
Cell Rule → CNF

A number appears once in each cell

There is at least one number in each cell  (definedness)

\[ Cell_d' = \bigcup_{r=1}^{N} \bigcup_{c=1}^{N} \{ \mathcal{C}^N_{v=1} (r, c, v) \mid (r, c, v) \in V \} \]

There is at most one number in each cell  (uniqueness)
Example

\[
\begin{array}{|c|c|c|c|}
\hline
(1,1,2) & (1,2,1) & 4 & 3 \\
\hline
(2,1,2) & (2,2,1) & (2,3,1) & (2,4,1) \\
(2,1,3) & (2,2,2) & (2,3,2) & (2,4,2) \\
(2,1,4) & (2,2,4) & (2,3,3) & (2,4,3) \\
\hline
(3,1,2) & (3,2,2) & (3,3,1) & (3,4,1) \\
(3,1,4) & (3,2,4) & (3,3,2) & (3,4,2) \\
\hline
1 & 3 & (4,3,2) & (4,4,2) \\
\hline
\end{array}
\]

\[
Cell_{d}' = \left\{ (1,1,2) \right\} \\
\left\{ (1,2,1) \lor (1,2,2) \right\} \\
\left\{ (2,1,2) \lor (2,1,3) \lor (2,1,4) \right\} \\
\left\{ (2,2,1) \lor (2,2,2) \lor (2,2,4) \right\} \\
\cdots \\
\left\{ (4,3,2) \right\} \\
\left\{ (4,4,2) \lor (4,4,4) \right\}
\]

\[
Cell_{u}' = \left\{ \neg(1,2,1) \lor \neg(1,2,2) \right\} \\
\left\{ \neg(2,1,2) \lor \neg(2,1,3) \right\} \\
\left\{ \neg(2,1,2) \lor \neg(2,1,4) \right\} \\
\left\{ \neg(2,1,3) \lor \neg(2,1,4) \right\} \\
\cdots \\
\left\{ \neg(4,4,2) \lor \neg(4,4,4) \right\}
\]
Row Rule $\rightarrow$ CNF

Each number appears at least in each row (definedness)

$$Row_d' = \bigcup_{r=1}^{N} \bigcup_{v=1}^{N} \{ i_{c=1}^N (r, c, v) | (r, c, v) \in V \}$$

Each number appears at most in each row (uniqueness)
**Example**

<table>
<thead>
<tr>
<th></th>
<th>(1,2,1)</th>
<th>(1,2,2)</th>
<th>4</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1,1,2)</td>
<td>(1,1,2)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(2,1,2)</td>
<td>(2,2,1)</td>
<td>(2,3,1)</td>
<td>(2,4,1)</td>
<td></td>
</tr>
<tr>
<td>(2,1,3)</td>
<td>(2,2,2)</td>
<td>(2,3,2)</td>
<td>(2,4,2)</td>
<td></td>
</tr>
<tr>
<td>(2,1,4)</td>
<td>(2,2,4)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(3,1,2)</td>
<td>(3,2,2)</td>
<td>(3,3,1)</td>
<td>(3,4,1)</td>
<td></td>
</tr>
<tr>
<td>(3,1,3)</td>
<td>(3,2,4)</td>
<td>(3,3,2)</td>
<td>(3,4,2)</td>
<td></td>
</tr>
<tr>
<td>(3,1,4)</td>
<td></td>
<td>(3,3,3)</td>
<td>(3,4,3)</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>3</td>
<td>(4,3,2)</td>
<td>(4,4,2)</td>
<td></td>
</tr>
</tbody>
</table>

\[
\text{Row}_{d}^{' = \{ (1,2,1) } \\
\{ (1,1,2) \lor (1,2,2) \} \\
\{ (2,2,1) \lor (2,3,1) \lor (2,4,1) \} \\
\{ (2,1,2) \lor (2,2,2) \lor (2,3,2) \lor (2,4,2) \} \\
\ldots \\
\{ (4,3,2) \lor (4,4,2) \} \\
\{ (4,4,4) \} \\
\]

\[
\text{Row}_{u}^{' = \{ \neg (1,1,2) \lor \neg (1,2,2) \} \\
\{ \neg (2,2,1) \lor \neg (2,3,1) \} \\
\{ \neg (2,1,2) \lor \neg (2,4,1) \} \\
\{ \neg (2,3,1) \lor \neg (2,4,1) \} \\
\ldots \\
\{ \neg (4,3,2) \lor \neg (4,4,2) \} \\
\} \\
\]
Column Rule $\rightarrow$ CNF

A number appears once in each column

Each number appears at least in each column (definedness)

$$Col_d^d = \bigcup_{c=1}^{N} \bigcap_{v=1}^{N} \{ c^N_{r=1} (r, c, v) | (r, c, v) \in V \}$$

Each number appears at most in each column (uniqueness)
Example

\[
\begin{array}{cccc}
(1,1,2) & (1,2,1) & 4 & 3 \\
(2,1,2) & (2,2,1) & (2,3,1) & (2,4,1) \\
(2,1,3) & (2,2,2) & (2,3,2) & (2,4,2) \\
(2,1,4) & (2,2,4) & (2,3,3) & (2,4,3) \\
(3,1,2) & (3,2,2) & (3,3,1) & (3,4,1) \\
(3,1,3) & (3,2,3) & (3,3,2) & (3,4,2) \\
(3,1,4) & (3,2,4) & (3,3,3) & (3,4,3) \\
1 & 3 & (4,3,2) & (4,4,2) \\
& & (4,4,2) & (4,4,4) \\
\end{array}
\]

\[Col_d' = \begin{cases} 
(1,1,2) \lor (2,1,2) \lor (3,1,2) \\
(2,1,3) \\
(2,1,4) \lor (3,1,4) \\
\ldots \\
(2,4,2) \lor (3,4,2) \lor (4,4,2) \\
(3,4,4) \lor (4,4,4)
\end{cases}\]

\[Col_u' = \begin{cases} 
\neg(1,1,2) \lor \neg(2,1,2) \\
\neg(1,1,2) \lor \neg(3,1,2) \\
\neg(2,1,4) \lor \neg(3,1,4) \\
\ldots \\
\neg(3,4,4) \lor \neg(4,4,4)
\end{cases}\]
Block Rule $\Rightarrow$ CNF

A number appears once in each block

Each number appears at least in each block (definedness)

Each number appears at most in each block (uniqueness)
Example

<table>
<thead>
<tr>
<th></th>
<th>(1,2,1)</th>
<th>(1,2,2)</th>
<th>4</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1,1,2)</td>
<td>(1,2,1)</td>
<td>(1,2,2)</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>(2,1,2)</td>
<td>(2,2,1)</td>
<td>(2,3,1)</td>
<td>(2,4,1)</td>
<td></td>
</tr>
<tr>
<td>(2,1,3)</td>
<td>(2,2,2)</td>
<td>(2,3,2)</td>
<td>(2,4,2)</td>
<td></td>
</tr>
<tr>
<td>(2,1,4)</td>
<td>(2,2,4)</td>
<td>(2,3,3)</td>
<td>(2,4,4)</td>
<td></td>
</tr>
<tr>
<td>(3,1,2)</td>
<td>(3,2,1)</td>
<td>(3,3,1)</td>
<td>(3,4,1)</td>
<td></td>
</tr>
<tr>
<td>(3,1,3)</td>
<td>(3,2,2)</td>
<td>(3,3,2)</td>
<td>(3,4,2)</td>
<td></td>
</tr>
<tr>
<td>(3,1,4)</td>
<td>(3,2,4)</td>
<td>(3,3,3)</td>
<td>(3,4,3)</td>
<td></td>
</tr>
<tr>
<td>(3,2,1)</td>
<td>(3,2,2)</td>
<td>(3,3,1)</td>
<td>(3,4,1)</td>
<td></td>
</tr>
<tr>
<td>(3,2,2)</td>
<td>(3,2,3)</td>
<td>(3,3,2)</td>
<td>(3,4,2)</td>
<td></td>
</tr>
<tr>
<td>(3,2,3)</td>
<td>(3,2,4)</td>
<td>(3,3,3)</td>
<td>(3,4,3)</td>
<td></td>
</tr>
<tr>
<td>(3,2,4)</td>
<td>(3,3,2)</td>
<td>(3,3,3)</td>
<td>(3,4,4)</td>
<td></td>
</tr>
</tbody>
</table>

Block $d' = \{(1,2,1) \lor (2,2,1)\}
\{(1,1,2) \lor (1,2,2) \lor (2,1,2) \lor (2,2,2)\}
\{(2,1,3)\}
\{\ldots\}
\{\ldots\}
\{\ldots\}
\{(3,3,3)\}
\{(3,4,4) \lor (4,4,4)\}$

Block $u' = \{\neg (1,2,1) \lor \neg (2,2,1)\}
\{\neg (1,1,2) \lor \neg (1,2,2)\}
\{\neg (1,1,2) \lor \neg (2,1,2)\}
\{\neg (1,1,2) \lor \neg (2,2,2)\}
\{\ldots\}
\{\ldots\}
\{\ldots\}
\{\neg (3,4,4) \lor \neg (4,4,4)\}$
Optimized Encoding

The resulting CNF formula

$$\phi = \text{Cell}_d' \cup \text{Cell}_u' \cup \text{Row}_d' \cup \text{Row}_u' \cup \text{Col}_d' \cup \text{Col}_u' \cup \text{Block}_d' \cup \text{Block}_u'$$

$$\phi$$ is **satisfiable** iff Sudoku has a **solution**

**Smaller** variables and clauses than previous encodings

Number of variables are reduced 12 times on average in our experiments

Number of clauses are reduced 79 times on average in our experiments
Agenda

• Introduction

• Background and Previous Encodings

• Optimized Encoding

• Experimental Results

• Conclusions
## Experimental Results

<table>
<thead>
<tr>
<th>size</th>
<th>level</th>
<th>vars</th>
<th>clauses</th>
<th>time</th>
<th>vars</th>
<th>clauses</th>
<th>time</th>
<th>k</th>
<th>ratio</th>
<th>vars↓</th>
<th>claus↓</th>
</tr>
</thead>
<tbody>
<tr>
<td>9x9</td>
<td>easy</td>
<td>729</td>
<td>12013</td>
<td>0.00</td>
<td>220</td>
<td>1761</td>
<td>0.00</td>
<td>26</td>
<td>32</td>
<td>3</td>
<td>7</td>
</tr>
<tr>
<td>9x9</td>
<td>hard</td>
<td>729</td>
<td>12018</td>
<td>0.00</td>
<td>164</td>
<td>1070</td>
<td>0.00</td>
<td>30</td>
<td>37</td>
<td>5</td>
<td>11</td>
</tr>
<tr>
<td>16x16</td>
<td>easy</td>
<td>4096</td>
<td>124008</td>
<td>0.01</td>
<td>648</td>
<td>5598</td>
<td>0.00</td>
<td>104</td>
<td>41</td>
<td>6</td>
<td>22</td>
</tr>
<tr>
<td>16x16</td>
<td>hard</td>
<td>4096</td>
<td>124002</td>
<td>0.01</td>
<td>797</td>
<td>8552</td>
<td>0.00</td>
<td>98</td>
<td>38</td>
<td>5</td>
<td>15</td>
</tr>
<tr>
<td>25x25</td>
<td>easy</td>
<td>15625</td>
<td>752792</td>
<td>0.07</td>
<td>1762</td>
<td>19657</td>
<td>0.04</td>
<td>292</td>
<td>47</td>
<td>9</td>
<td>38</td>
</tr>
<tr>
<td>25x25</td>
<td>hard</td>
<td>15625</td>
<td>752778</td>
<td>0.21</td>
<td>1990</td>
<td>24137</td>
<td>0.05</td>
<td>278</td>
<td>45</td>
<td>8</td>
<td>31</td>
</tr>
<tr>
<td>36x36</td>
<td>easy</td>
<td>46656</td>
<td>3271748</td>
<td>0.50</td>
<td>4186</td>
<td>57595</td>
<td>0.06</td>
<td>644</td>
<td>50</td>
<td>11</td>
<td>57</td>
</tr>
<tr>
<td>36x36</td>
<td>hard</td>
<td>46656</td>
<td>327168</td>
<td>0.67</td>
<td>3673</td>
<td>45383</td>
<td>0.08</td>
<td>664</td>
<td>51</td>
<td>13</td>
<td>72</td>
</tr>
<tr>
<td>49x49</td>
<td>easy</td>
<td>117649</td>
<td>11305189</td>
<td>1.47</td>
<td>7642</td>
<td>112444</td>
<td>0.13</td>
<td>1282</td>
<td>53</td>
<td>15</td>
<td>101</td>
</tr>
<tr>
<td>64x64</td>
<td>easy</td>
<td>262144</td>
<td>33048912</td>
<td>stack</td>
<td>11440</td>
<td>169772</td>
<td>0.04</td>
<td>2384</td>
<td>58</td>
<td>23</td>
<td>195</td>
</tr>
<tr>
<td>81x81</td>
<td>easy</td>
<td>531441</td>
<td>85060787</td>
<td>stack</td>
<td>17793</td>
<td>266025</td>
<td>0.06</td>
<td>3983</td>
<td>61</td>
<td>30</td>
<td>320</td>
</tr>
</tbody>
</table>
81x81 Puzzle

Variables are reduced 30 times

Clauses are reduced 320 times
Variable Reduction

![Graph showing variable reduction for different sizes. The x-axis represents the size of the variable, and the y-axis represents the number of variables. The graph compares the number of variables using the extended and proposed methods.](image-url)
Clause Reduction

The graph shows the number of clauses for different sizes (9x9, 16x16, 25x25, 36x36, 49x49, 64x64, 81x81) for two methods: extended (blue) and proposed (red). The y-axis represents the number of clauses, while the x-axis represents the size of the input. The proposed method significantly reduces the number of clauses, especially for larger sizes.
Time Reduction
Variable Reduction Ratio

The graph shows the relationship between the variable reduction ratio and the percentage k. As k increases, the variable reduction ratio also increases significantly, indicating a strong correlation between the two variables.
Clause Reduction Ratio

![Graph showing the clause reduction ratio against k (%)]

- **x-axis**: k (%)
- **y-axis**: clause reduction

The graph demonstrates how the clause reduction ratio increases significantly as k (%) increases from 32 to 61.
Agenda

• Introduction

• Background and Previous Encodings

• Optimized Encoding

• Experimental Results

• Conclusions
Conclusions

Previous encodings

J. Ouaknine, Sudoku as a SAT Problem, 2006
T. Weber, A SAT-based Sudoku Solver, 2005

Props and cons

+ Ideal encoding techniques
+ Well used for small puzzles
  – Too many clauses
  – Hard to handle large size puzzles such as 81x81
Conclusions

Proposed techniques

Optimized encoding used to reduce a formula

Results from 11 different size puzzles

+ All given puzzles are successfully solved
+ Number of variables is greatly reduced
+ Number of clauses is greatly reduced
+ Execution time is greatly reduced
+ Finally, encoding time is greatly reduced

Thank You!!