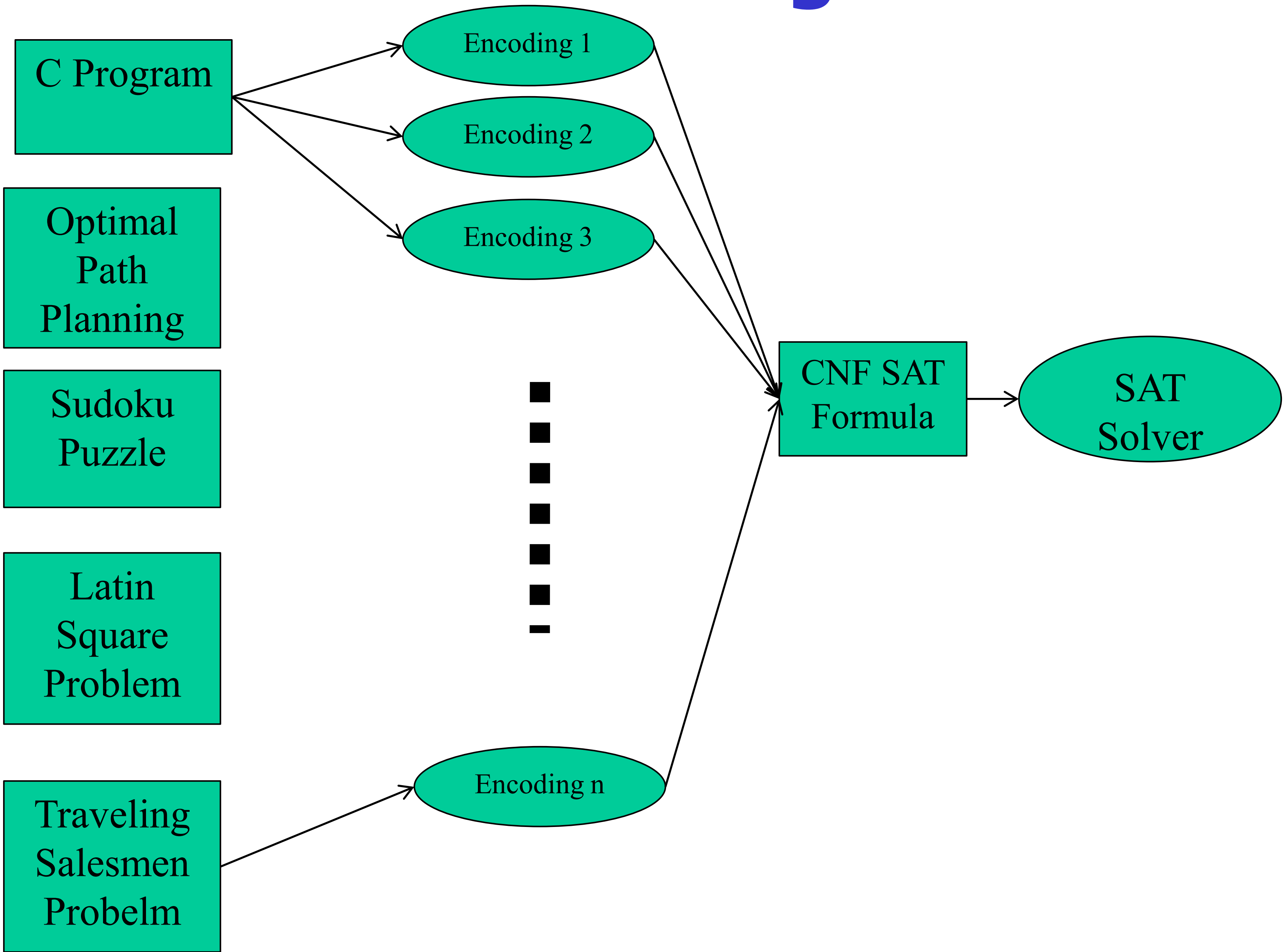
SAT Encodings for Sudoku

Bug Catching in 2006 Fall

Sep. 26, 2006

Gi-Hwon Kwon

Various SAT Encoding



Agenda

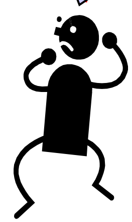
- **Introduction**
- Background and Previous Encodings
- Optimized Encoding
- Experimental Results
- Conclusions

What is Sudoku ?

Problem

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

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



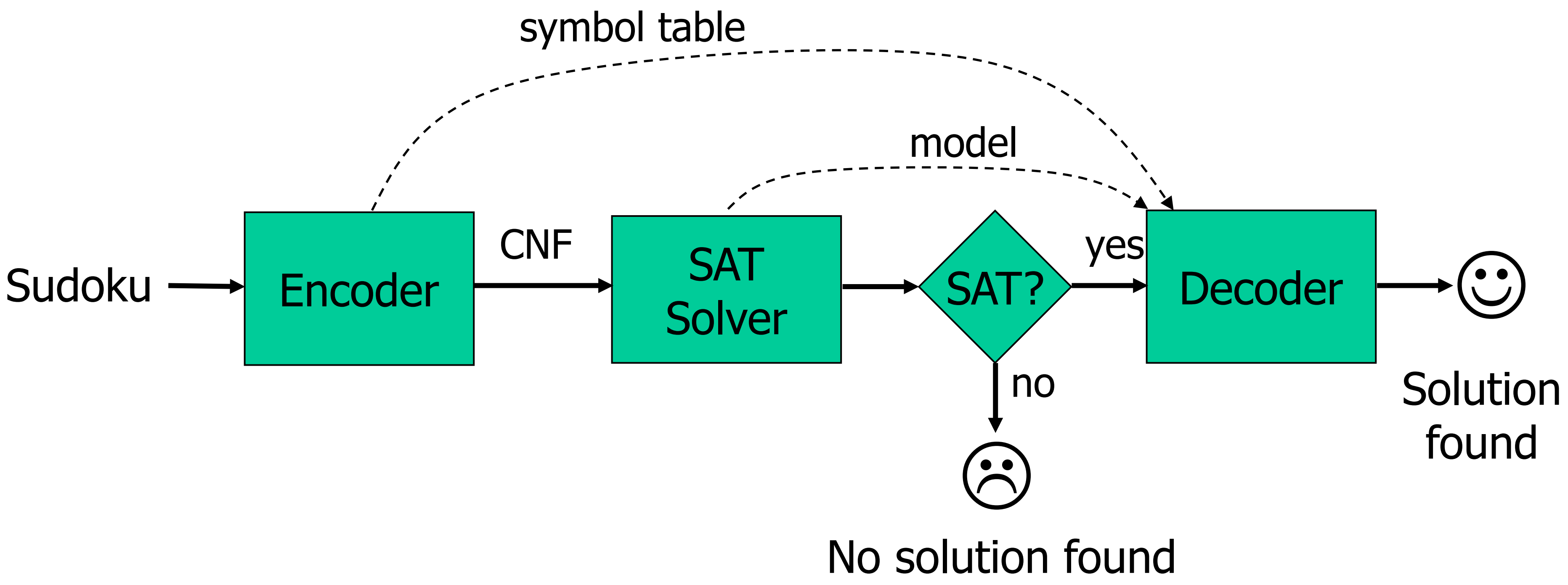
Solution

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

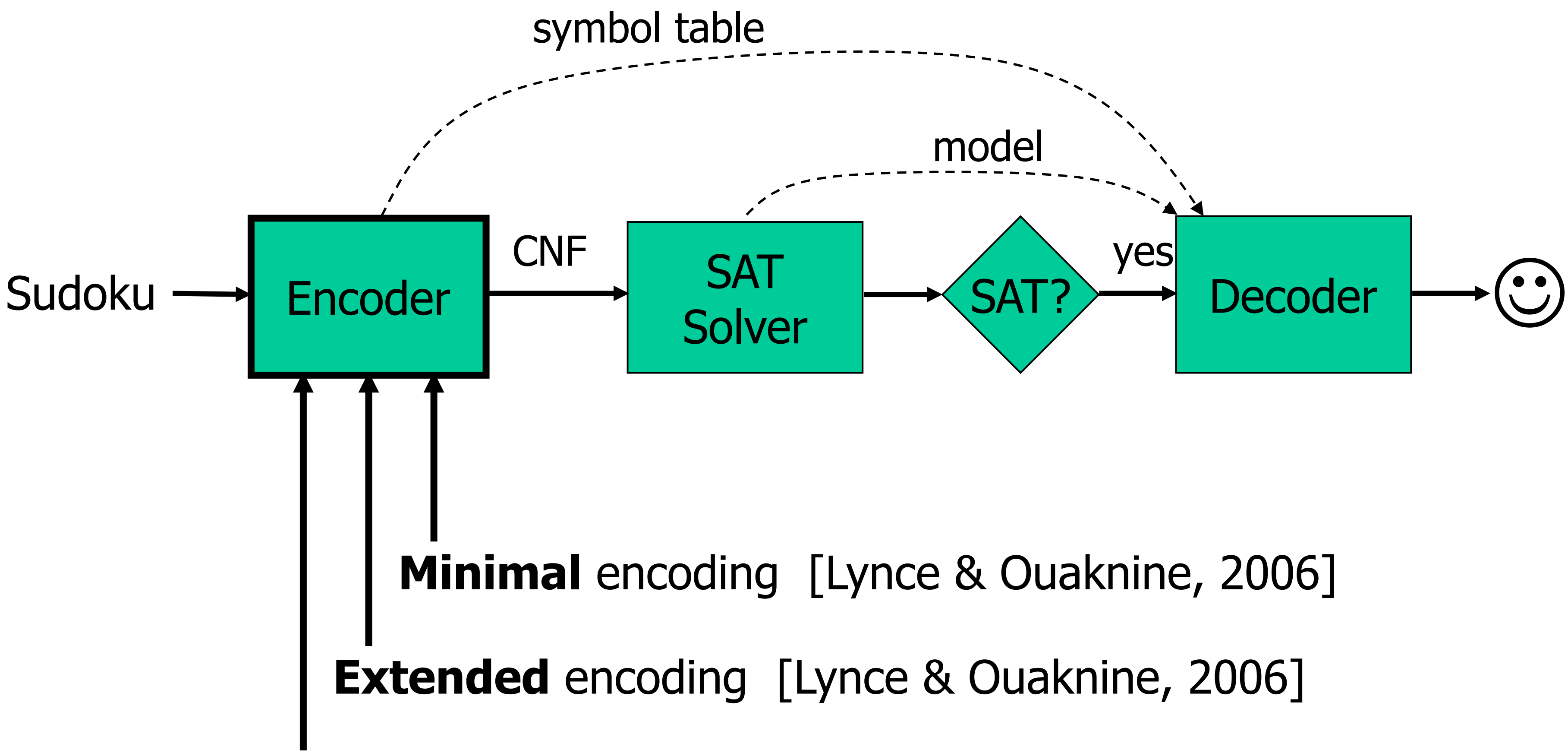
Sodoku rules

- ✓ 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



Previous Encodings



Minimal encoding [Lynce & Ouaknine, 2006]

Extended encoding [Lynce & Ouaknine, 2006]

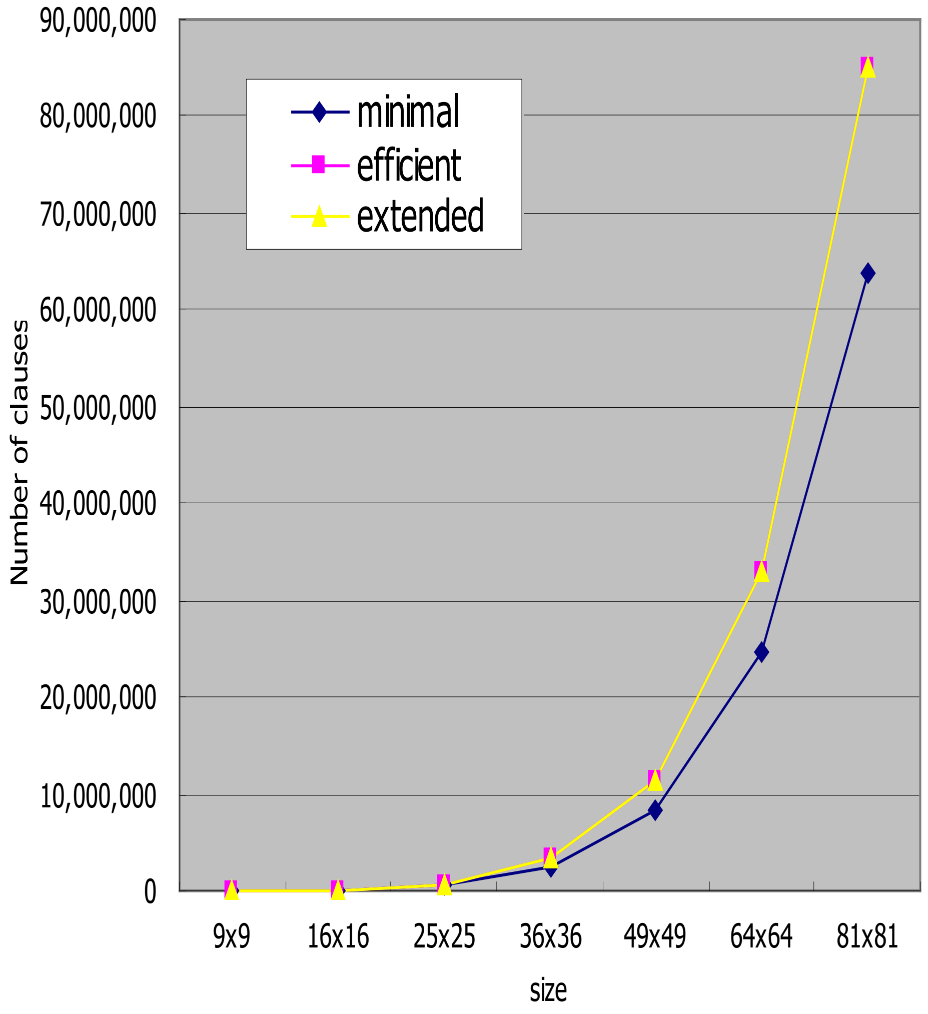
Efficient encoding [Weber, 2005]

Analysis of Previous Encodings

Encoding	Number of Variables	Number of Clauses
Minimal	N^3	$N*N+\left(N*N*\left(\frac{N*(N-1)}{2}\right)\right)*3+k$
Efficient	N^3	$N*N+\left(N*N*\left(\frac{N*(N-1)}{2}\right)\right)*4+k$
Extended	N^3	$\left(N*N+N*N*\left(\frac{N*(N-1)}{2}\right)\right)*4+k$

Exponential Growth in Clauses

size	minimal	efficient	extended
9x9	8829	11745	11988
16x16	92416	123136	123904
25x25	563125	750625	752500
36x36	2450736	3267216	3271104
49x49	8473129	11296705	11303908
64x64	24776704	33034240	33046528
81x81	63779481	85037121	85056804



Experimental Results

		minimal encoding			efficient encoding			extended encoding		
size	level	vars	clauses	time	vars	clauses	time	vars	clauses	time
9x9	easy	729	8854	0.00	729	11770	0.00	729	12013	0.00
9x9	hard	729	8859	0.00	729	11775	0.00	729	12018	0.00
16x16	easy	4096	92520	0.10	4096	123240	0.09	4096	124008	0.01
16x16	hard	4096	92514	0.46	4096	123234	0.21	4096	124002	0.01
25x25	easy	15625	563417	9.07	15625	750917	17.48	15625	752792	0.07
25x25	hard	15625	563403	time	15625	750903	time	15625	752778	0.21
36x36	easy	46656	2451380	time	46656	3267860	time	46656	3271748	0.50
36x36	hard	46656	2451400	time	46656	3267880	time	46656	3271768	0.67
49x49	easy	117649	8474410	time	117649	11297986	time	117649	11305189	1.47
64x64	easy	262144	24779088	stack	262144	33036624	stack	262144	33048912	stack
81x81	easy	531441	63783464	stack	531441	85041104	stack	531441	85060787	stack

Experimental Results

		minimal encoding			efficient encoding			extended encoding		
size	level	vars	clauses	time	vars	clauses	time	vars	clauses	time
9x9	easy	729	8854	0.00	729	11770	[illegible]	29	12013	0.00
9x9	hard	729	8859	0.00	729	11775	[illegible]	29	12018	0.00
16x16	easy	4096	92520	0.10	4096	123240	[illegible]	96	124008	0.01
16x16	hard	4096	92514	0.46	4096	[illegible]	[illegible]	[illegible]	24002	0.01
25x25	easy	15625	563417	9.07	15625	750917	17.48	15625	752792	0.07
25x25	hard	15625	563403	time	15625	750903	time	15625	752778	0.21
36x36	easy	46656	2451380	[illegible]	5656	3267860	time	46656	3271748	0.50
36x36	hard	46656	2451400	[illegible]	5656	3267880	time	46656	3271768	0.67
49x49	easy	117649	8474410	[illegible]	7649	11297986	time	117649	11305189	1.47
64x64	easy	262144	[illegible]	[illegible]	[illegible]	6624	stack	262144	33048912	stack
81x81	easy	531441	[illegible]	stack	531441	85041104	stack	531441	85060787	stack

Solution found

No solution found

Motivations

- Sudoku was regarded as SAT problem
  - W Weber, A SAT-based Sudoku Solver, Nov. 2005.
  - 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

Agenda

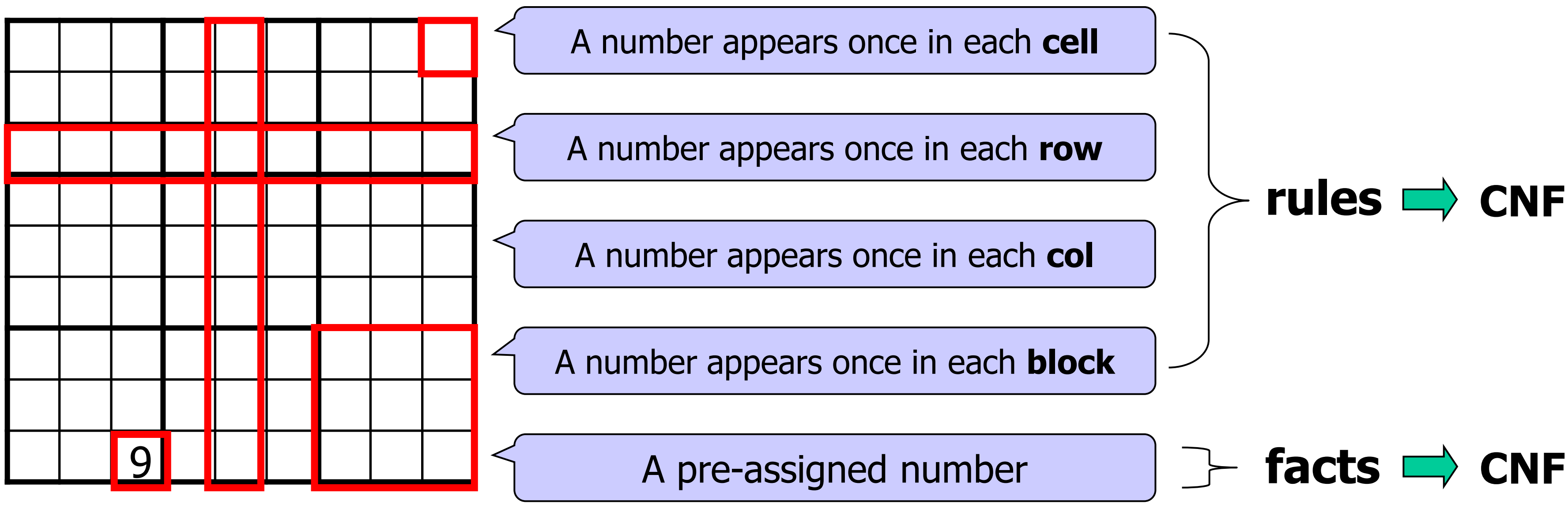
- Introduction
- **Background and Previous Encodings**
- Optimized Encoding
- Experimental Results
- Conclusions

Encoding

- Kowledge compilation into a **target language**

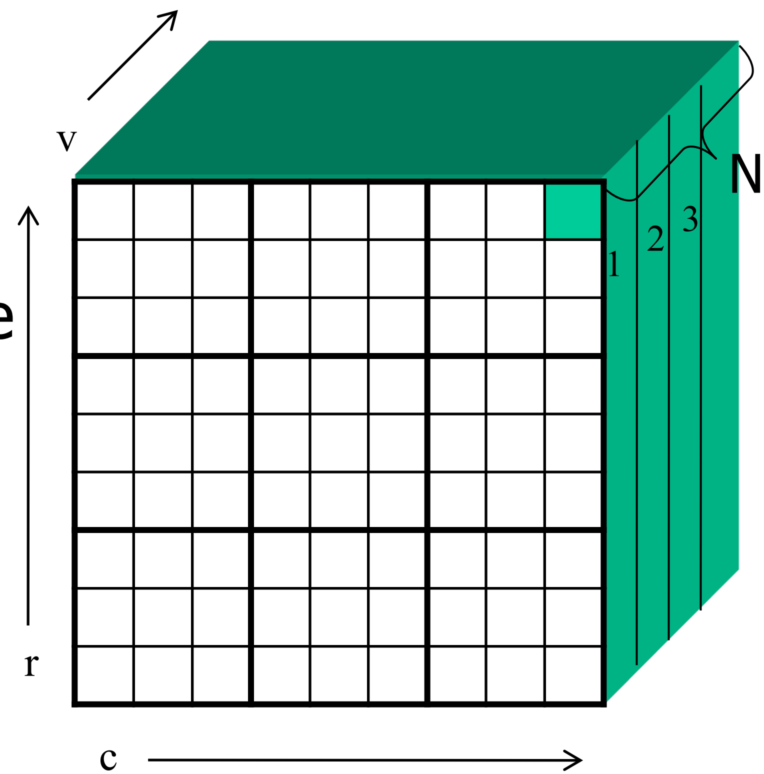
problem knowlege ➡ CNF

- Knowlede about Sudoku

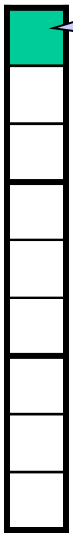


Variables

- 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) (i.e., x_{rcv}) iff [r,c] = v
  - $\neg$(r,c,v) (i.e.,$\neg x_{rcv}$) iff [r,c] $\neq$ v



Cell Rule → CNF



A number appears once in each **cell**

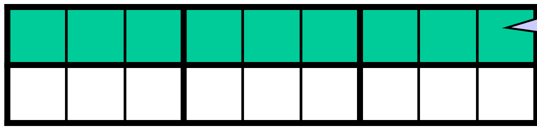
There is **at least** one number in each cell (definedness)

$$Cell_d = \wedge_{r=1}^{N} \wedge_{c=1}^{N} \vee_{v=1}^{N} (r, c, v)$$

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

$$Cell_u = \wedge_{r=1}^{N} \wedge_{c=1}^{N} \wedge_{v_i=1}^{N-1} \wedge_{v_j=v_i+1}^{N} \neg((r, c, v_i) \wedge (r, c, v_j))$$

Row Rule → CNF



A number appears once in each **row**

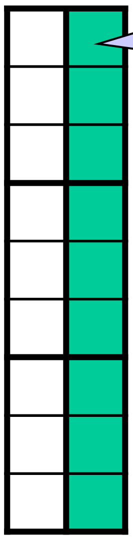
Each number appears **at least** once in each row (definedness)

$$Row_d = \wedge_{r=1}^{N} \wedge_{v=1}^{N} \vee_{c=1}^{N} (r,c,v)$$

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

$$Row_u = \wedge_{r=1}^{N} \wedge_{v=1}^{N} \wedge_{c_i=1}^{N-1} \wedge_{c_j=c_i+1}^{N} \neg((r,c_i,v) \wedge (r,c_j,v))$$

Column Rule → CNF



A number appears once in each **column**

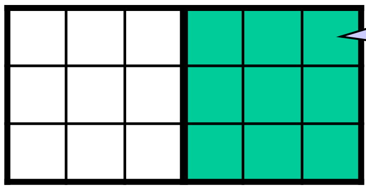
Each number appears **at least once** in each column (definedness)

$$Col_d = \wedge_{c=1}^{N} \wedge_{v=1}^{N} \vee_{r=1}^{N} (r,c,v)$$

Each number appears **at most** once in each column (uniqueness)

$$Col_u = \wedge_{c=1}^{N} \wedge_{v=1}^{N} \wedge_{r_i=1}^{N-1} \wedge_{r_j=r_i+1}^{N} \neg((r_i,c,v) \wedge (r_j,c,v))$$

Block Rule → CNF



A number appears once in each **block**

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

$$Block_d = \wedge_{r_{offs}=1}^{subN} \wedge_{c_{offs}=1}^{subN} \wedge_{v=1}^{N} \vee_{r=1}^{subN} \vee_{c=1}^{subN} ((r_{offs}-1)*subN + r, (c_{offs}-1)*subN + c, v)$$

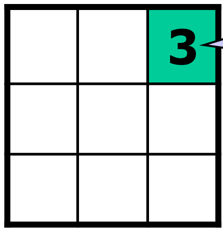
Each number appears **at most** once in each block (uniqueness)

$$Block_u = \wedge_{r_{offs}=1}^{subN} \wedge_{c_{offs}=1}^{subN} \wedge_{v=1}^{N} \wedge_{r=1}^{N} \wedge_{c=r+1}^{N}$$

$$\neg((r_{offs}-1)*subN + (r \bmod subN), (c_{offs}-1)*subN + (r \bmod subN), v)$$

$$\wedge ((r_{offs}-1)*subN + (c \bmod subN), (c_{offs}-1)*subN + (c \bmod subN), v))$$

Pre-Assigned Fact → CNF



A pre-assigned number

As a constant; the number is never changed

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

$$Assigned = \wedge_{i=1}^{k} \{(r,c,a) \mid \exists_{1\le a\le N} \bullet [r,c]=a\}$$

where k is a number of pre-assigned numbers

Previous Encodings

Minimal encoding [Lynce & Ouaknine, 2006]

$$\phi = Cell_d \cup Row_u \cup Col_u \cup Block_u \cup Assigned$$

<u>sufficient to characterize the puzzle</u>

Extended encoding [Lynce & Ouaknine, 2006]

$$\phi = Cell_d \cup Cell_u \cup Row_d \cup Row_u \cup Col_d \cup Col_u \cup Block_d \cup Block_u \cup Assigned$$

<u>minimal encoding with redundant clauses</u>

Efficient encoding [Weber, 2005]

$$\phi = Cell_d \cup Cell_u \cup Row_u \cup Col_u \cup Block_u \cup Assigned$$

<u>between minimal encoding and extended encoding</u>

Analysis (Recap)

Encoding	Number of Variables	Number of Clauses
Minimal	N^3	$N*N+\left(N*N*\left(\frac{N*(N-1)}{2}\right)\right)*3+k$
Efficient	N^3	$N*N+\left(N*N*\left(\frac{N*(N-1)}{2}\right)\right)*4+k$
Extended	N^3	$\left(N*N+N*N*\left(\frac{N*(N-1)}{2}\right)\right)*4+k$

Agenda

- Introduction
- Background and Previous Encodings
- **Optimized Encoding**
- Experimental Results
- Conclusions

Example

		4	3
1	3		

CNF

is represented using boolean variables

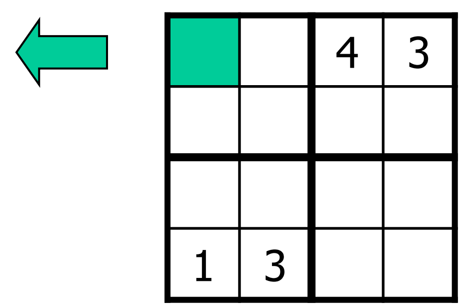
- For example, consider the cell [1,1]
  - Four cases are considered; thus, four variables are needed

(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 cosidered is [1,1]=2

(1,1,2)



Variables

- Let *V* be a set of variables

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

$$affected(r,c,v) = sameRow(r,c,v) \vee sameCol(r,c,v) \vee sameBlock(r,c,v)$$

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

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

$$sameBlock(r,c,v) = \exists_{i:\, originRow..subN} \bullet \exists_{i:\, originCol..subN} \bullet (i \neq r \wedge j \neq c) \Rightarrow [i,j] = v$$

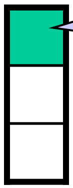
Example

$V =$

(1,1,2)	(1,2,1) (1,2,2)	4	3
(2,1,2) (2,1,3) (2,1,4)	(2,2,1) (2,2,2) (2,2,4)	(2,3,1) (2,3,2)	(2,4,1) (2,4,2)
(3,1,2) (3,1,4)	(3,2,2) (3,2,4)	(3,3,1) (3,3,2) (3,3,3)	(3,4,1) (3,4,2) (3,4,4)
1	3	(4,3,2)	(4,4,2) (4,4,4)

these parts are excluded

Cell Rule → CNF



A number appears once in each **cell**

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

$$Cell_d' = \bigcup_{r=1}^{N} \bigcup_{c=1}^{N} \{\vee_{v=1}^{N} (r,c,v) \mid (r,c,v) \in V\}$$

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

$$Cell_u' = \bigcup_{r=1}^{N} \bigcup_{c=1}^{N} \bigcup_{v_i=1}^{N-1} \bigcup_{v_j=v_i+1}^{N} \{\neg(r,c,v_i) \vee \neg(r,c,v_j)$$

$$\mid (r,c,v_i) \in V \wedge r,c,v_j) \in V\}$$

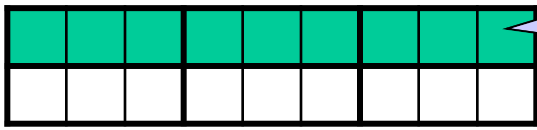
Example

(1,1,2)	(1,2,1) (1,2,2)	4	3
(2,1,2) (2,1,3) (2,1,4)	(2,2,1) (2,2,2) (2,2,4)	(2,3,1) (2,3,2)	(2,4,1) (2,4,2)
(3,1,2) (3,1,4)	(3,2,2) (3,2,4)	(3,3,1) (3,3,2) (3,3,3)	(3,4,1) (3,4,2) (3,4,4)
1	3	(4,3,2)	(4,4,2) (4,4,4)

$$Cell_d' = \left\{ \begin{array}{l} \{(1,1,2)\} \\ \{(1,2,1) \vee (1,2,2)\} \\ \{(2,1,2) \vee (2,1,3) \vee (2,1,4)\} \\ \{(2,2,1) \vee (2,2,2) \vee (2,2,4)\} \\ \ldots\ldots\ldots \\ \ldots\ldots\ldots \\ \{(4,3,2)\} \\ \{(4,4,2) \vee (4,4,4)\} \end{array} \right\}$$

$$Cell_u' = \left\{ \begin{array}{l} \{\neg(1,2,1) \vee \neg(1,2,2)\} \\ \{\neg(2,1,2) \vee \neg(2,1,3)\} \\ \{\neg(2,1,2) \vee \neg(2,1,4)\} \\ \{\neg(2,1,3) \vee \neg(2,1,4)\} \\ \ldots\ldots\ldots \\ \ldots\ldots\ldots \\ \{\neg(4,4,2) \vee \neg(4,4,4)\} \end{array} \right\}$$

Row Rule → CNF



A number appears once in each **row**

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

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

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

$$Row_u' = \bigcup_{r=1}^{N} \bigcup_{v=1}^{N} \bigcup_{c_i=1}^{N-1} \bigcup_{c_j=c_i+1}^{N} \{\neg(r,c_i,v) \vee \neg(r,c_j,v) \mid (r,c_i,v) \in V \wedge (r,c_j,v) \in V\}$$

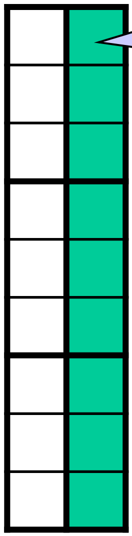
Example

(1,1,2)	(1,2,1) (1,2,2)	4	3
(2,1,2) (2,1,3) (2,1,4)	(2,2,1) (2,2,2) (2,2,4)	(2,3,1) (2,3,2)	(2,4,1) (2,4,2)
(3,1,2) (3,1,4)	(3,2,2) (3,2,4)	(3,3,1) (3,3,2) (3,3,3)	(3,4,1) (3,4,2) (3,4,4)
1	3	(4,3,2)	(4,4,2) (4,4,4)

$$Row_d' = \left\{ \begin{array}{l} \{(1,2,1)\} \\ \{(1,1,2) \vee (1,2,2)\} \\ \{(2,2,1) \vee (2,3,1) \vee (2,4,1)\} \\ \{(2,1,2) \vee (2,2,2) \vee (2,3,2) \vee (2,4,2)\} \\ \ldots\ldots\ldots \\ \ldots\ldots\ldots \\ \{(4,3,2) \vee (4,4,2)\} \\ \{(4,4,4)\} \end{array} \right\}$$

$$Row_u' = \left\{ \begin{array}{l} \{\neg(1,1,2) \vee \neg(1,2,2)\} \\ \{\neg(2,2,1) \vee \neg(2,3,1)\} \\ \{\neg(2,2,1) \vee \neg(2,4,1)\} \\ \{\neg(2,3,1) \vee \neg(2,4,1)\} \\ \ldots\ldots\ldots \\ \ldots\ldots\ldots \\ \{\neg(4,3,2) \vee \neg(4,4,2)\} \end{array} \right\}$$

Column Rule → CNF



A number appears once in each **column**

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

$$Col_d' = \bigcup_{c=1}^{N} \bigcup_{v=1}^{N} \{\vee_{r=1}^{N}(r,c,v) \mid (r,c,v) \in V\}$$

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

$$Col_u' = \bigcup_{c=1}^{N} \bigcup_{v=1}^{N} \bigcup_{r_i=1}^{N-1} \bigcup_{r_j=r_i+1}^{N} \{\neg(r_i,c,v) \vee \neg(r_j,c,v) \mid (r_i,c,v) \in V \wedge (r_j,c,v) \in V\}$$

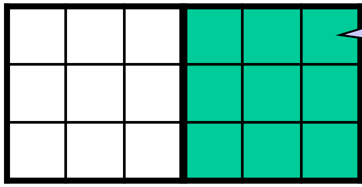
Example

(1,1,2)	(1,2,1) (1,2,2)	4	3
(2,1,2) (2,1,3) (2,1,4)	(2,2,1) (2,2,2) (2,2,4)	(2,3,1) (2,3,2)	(2,4,1) (2,4,2)
(3,1,2) (3,1,4)	(3,2,2) (3,2,4)	(3,3,1) (3,3,2) (3,3,3)	(3,4,1) (3,4,2) (3,4,4)
1	3	(4,3,2)	(4,4,2) (4,4,4)

$$Col_d' = \left\{ \begin{array}{l} \{(1,1,2) \vee (2,1,2) \vee (3,1,2)\} \\ \{(2,1,3)\} \\ \{(2,1,4) \vee (3,1,4)\} \\ \ldots\ldots\ldots \\ \ldots\ldots\ldots \\ \ldots\ldots\ldots \\ \{(2,4,2) \vee (3,4,2) \vee (4,4,2)\} \\ \{(3,4,4) \vee (4,4,4)\} \end{array} \right\}$$

$$Col_u' = \left\{ \begin{array}{l} \{\neg(1,1,2) \vee \neg(2,1,2)\} \\ \{\neg(1,1,2) \vee \neg(3,1,2)\} \\ \{\neg(2,1,2) \vee \neg(3,1,2)\} \\ \{\neg(2,1,4) \vee \neg(3,1,4)\} \\ \ldots\ldots\ldots \\ \ldots\ldots\ldots \\ \{\neg(3,4,4) \vee \neg(4,4,4)\} \end{array} \right\}$$

Block Rule → CNF



A number appears once in each **block**

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

$$Block_d' = \bigcup_{r_{offs}=1}^{subN} \bigcup_{c_{offs}=1}^{subN} \bigcup_{v=1}^{N} \{ \vee_{r=1}^{subN} \vee_{c=1}^{subN} (r_{offs} * subN + r, c_{offs} * subN + c, v)$$
$$| (r_{offs} * subN + r, c_{offs} * subN + c, v) \in V \}$$

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

$$Block_u' = \bigcup_{r_{offs}=1}^{subN} \bigcup_{c_{offs}=1}^{subN} \bigcup_{v=1}^{N} \bigcup_{r=1}^{N} \bigcup_{c=r+1}^{N}$$
$$\{ \neg (r_{offs} * subN + (r \ \mathrm{mod}\ subN), c_{offs} * subN + (r \ \mathrm{mod}\ subN), v)$$
$$\vee \neg (r_{offs} * subN + (c \ \mathrm{mod}\ subN), c_{offs} * subN + (c \ \mathrm{mod}\ subN), v)$$
$$| (r_{offs} * subN + (r \ \mathrm{mod}\ subN), c_{offs} * subN + (r \ \mathrm{mod}\ subN), v) \in V$$
$$\wedge (r_{offs} * subN + (c \ \mathrm{mod}\ subN), c_{offs} * subN + (c \ \mathrm{mod}\ subN), v) \in V \}$$

Example

(1,1,2)	(1,2,1) (1,2,2)	4	3
(2,1,2) (2,1,3) (2,1,4)	(2,2,1) (2,2,2) (2,2,4)	(2,3,1) (2,3,2)	(2,4,1) (2,4,2)
(3,1,2) (3,1,4)	(3,2,2) (3,2,4)	(3,3,1) (3,3,2) (3,3,3)	(3,4,1) (3,4,2) (3,4,4)
1	3	(4,3,2)	(4,4,2) (4,4,4)

$$Block_d' = \left\{ \begin{array}{l} \{(1,2,1) \vee (2,2,1)\} \\ \{(1,1,2) \vee (1,2,2) \vee (2,1,2) \vee (2,2,2)\} \\ \{(2,1,3)\} \\ \ldots\ldots\ldots \\ \ldots\ldots\ldots \\ \ldots\ldots\ldots \\ \{(3,3,3)\} \\ \{(3,4,4) \vee (4,4,4)\} \end{array} \right\}$$

$$Block_u' = \left\{ \begin{array}{l} \{\neg(1,2,1) \vee \neg(2,2,1)\} \\ \{\neg(1,1,2) \vee \neg(1,2,2)\} \\ \{\neg(1,1,2) \vee \neg(2,1,2)\} \\ \{\neg(1,1,2) \vee \neg(2,2,2)\} \\ \ldots\ldots\ldots \\ \ldots\ldots\ldots \\ \{\neg(3,4,4) \vee \neg(4,4,4)\} \end{array} \right\}$$

Optimized Encoding

The resulting CNF formula

$$\phi = Cell_d' \cup Cell_u' \cup Row_d' \cup Row_u' \cup Col_d' \cup Col_u' \cup Block_d' \cup Block_u'$$

ϕ 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

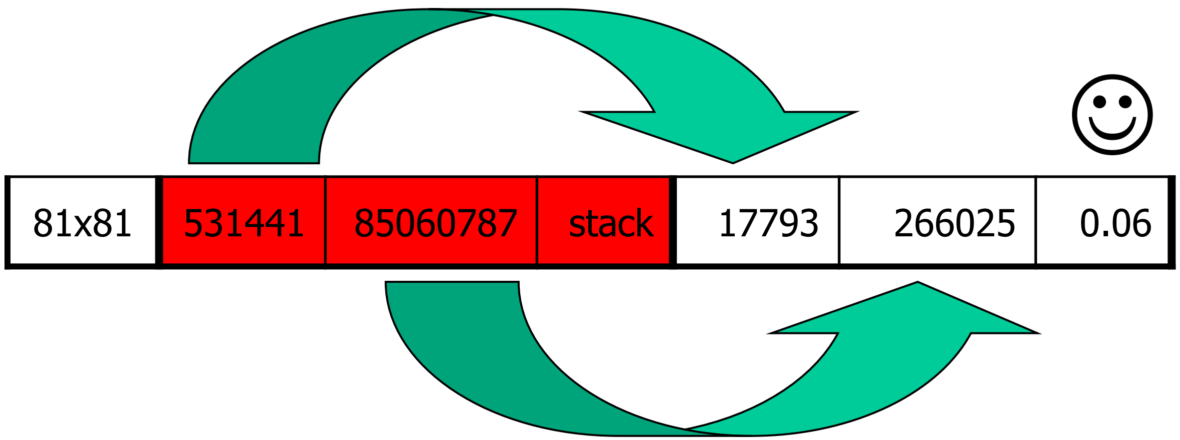
- Introduction
- Background and Previous Encodings
- Optimized Encoding
- **Experimental Results**
- Conclusions

Experimental Results

		extended encoding			proposed encoding			analysis of pre-assigned cells			
size	level	vars	clauses	time	vars	clauses	time	k	ratio	vars↓	claus↓
9x9	easy	729	12013	0.00	220	1761	0.00	26	32	3	7
9x9	hard	729	12018	0.00	164	1070	0.00	30	37	5	11
16x16	easy	4096	124008	0.01	648	5598	0.00	104	41	6	22
16x16	hard	4096	124002	0.01	797	8552	0.00	98	38	5	15
25x25	easy	15625	752792	0.07	1762	19657	0.04	292	47	9	38
25x25	hard	15625	752778	0.21	1990	24137	0.05	278	45	8	31
36x36	easy	46656	3271748	0.50	4186	57595	0.06	644	50	11	57
36x36	hard	46656	3271768	0.67	3673	45383	0.08	664	51	13	72
49x49	easy	117649	11305189	1.47	7642	112444	0.13	1282	53	15	101
64x64	easy	262144	33048912	stack	11440	169772	0.04	2384	58	23	195
81x81	easy	531441	85060787	stack	17793	266025	0.06	3983	61	30	320

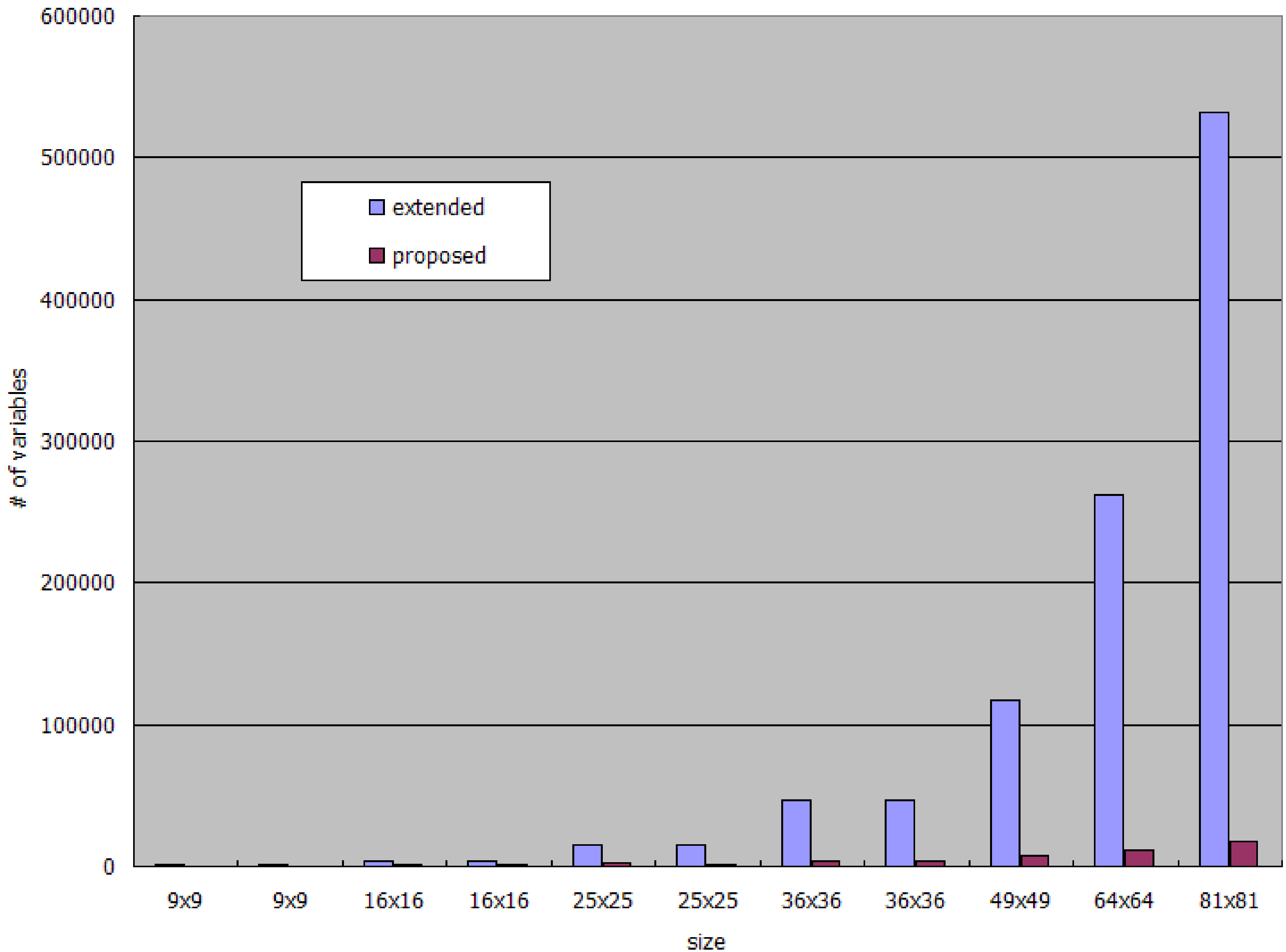
81x81 Puzzle

Variables are reduced 30 times

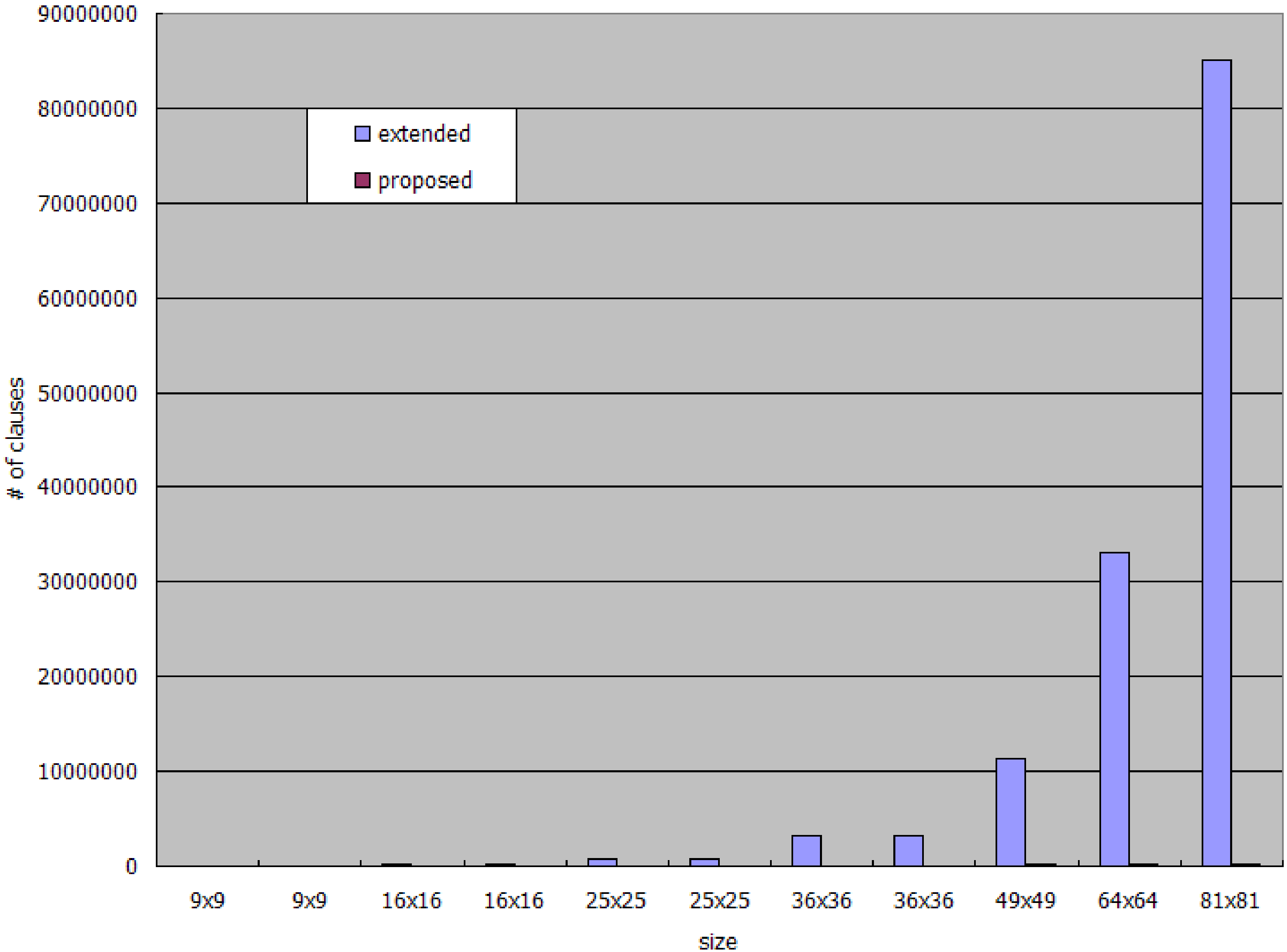


Clauses are reduced 320 times

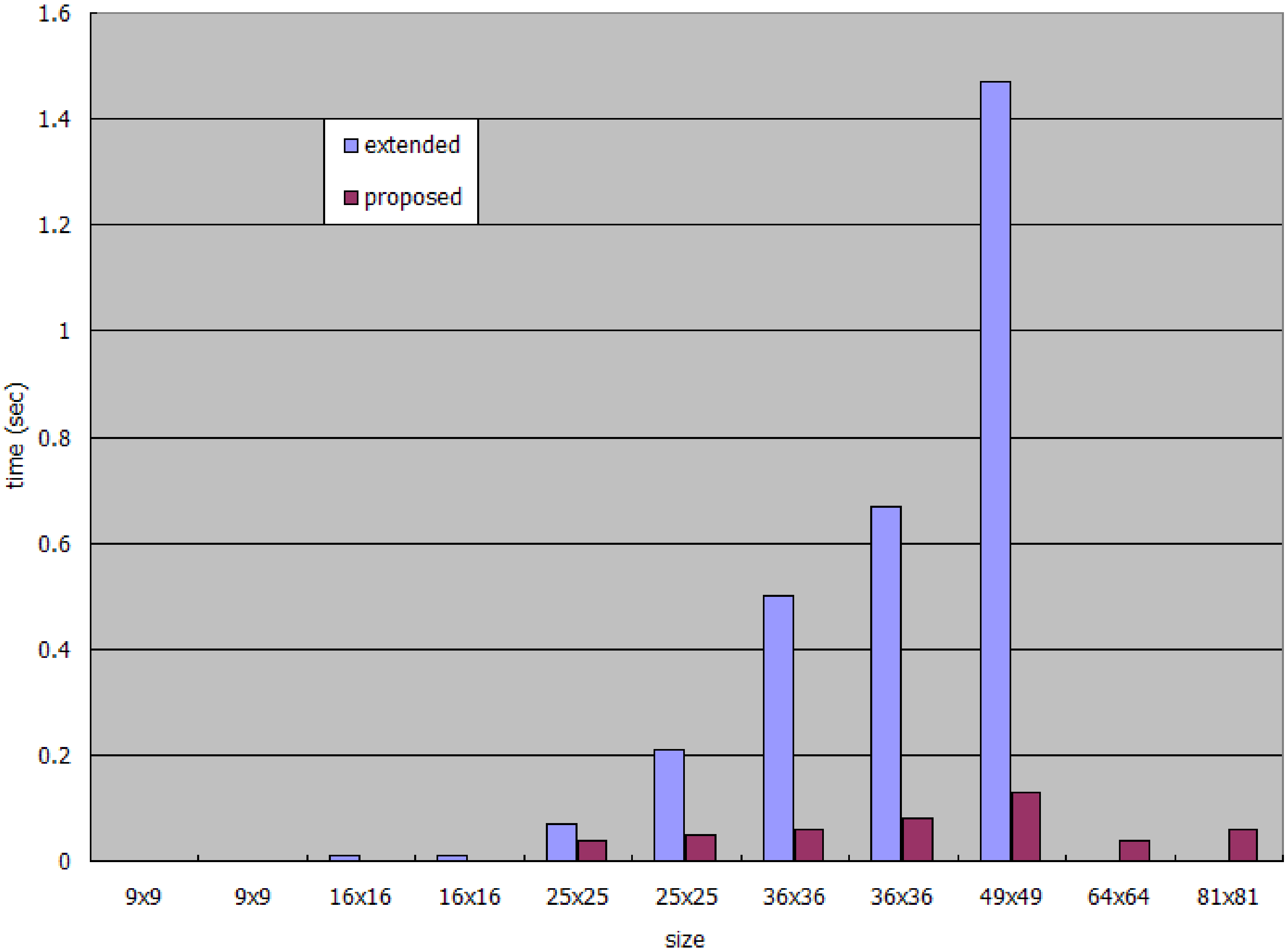
Variable Reduction



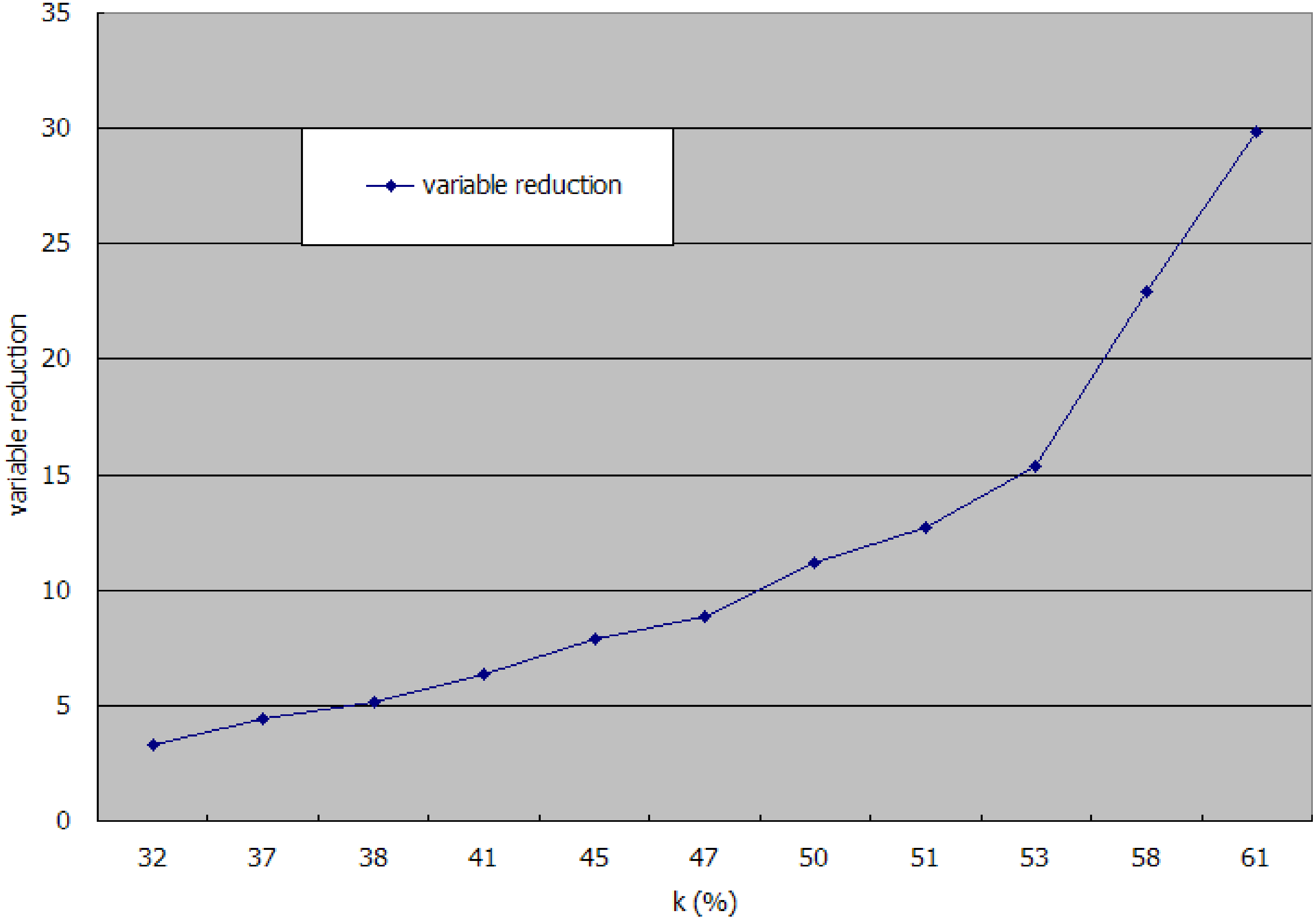
Clause Reduction



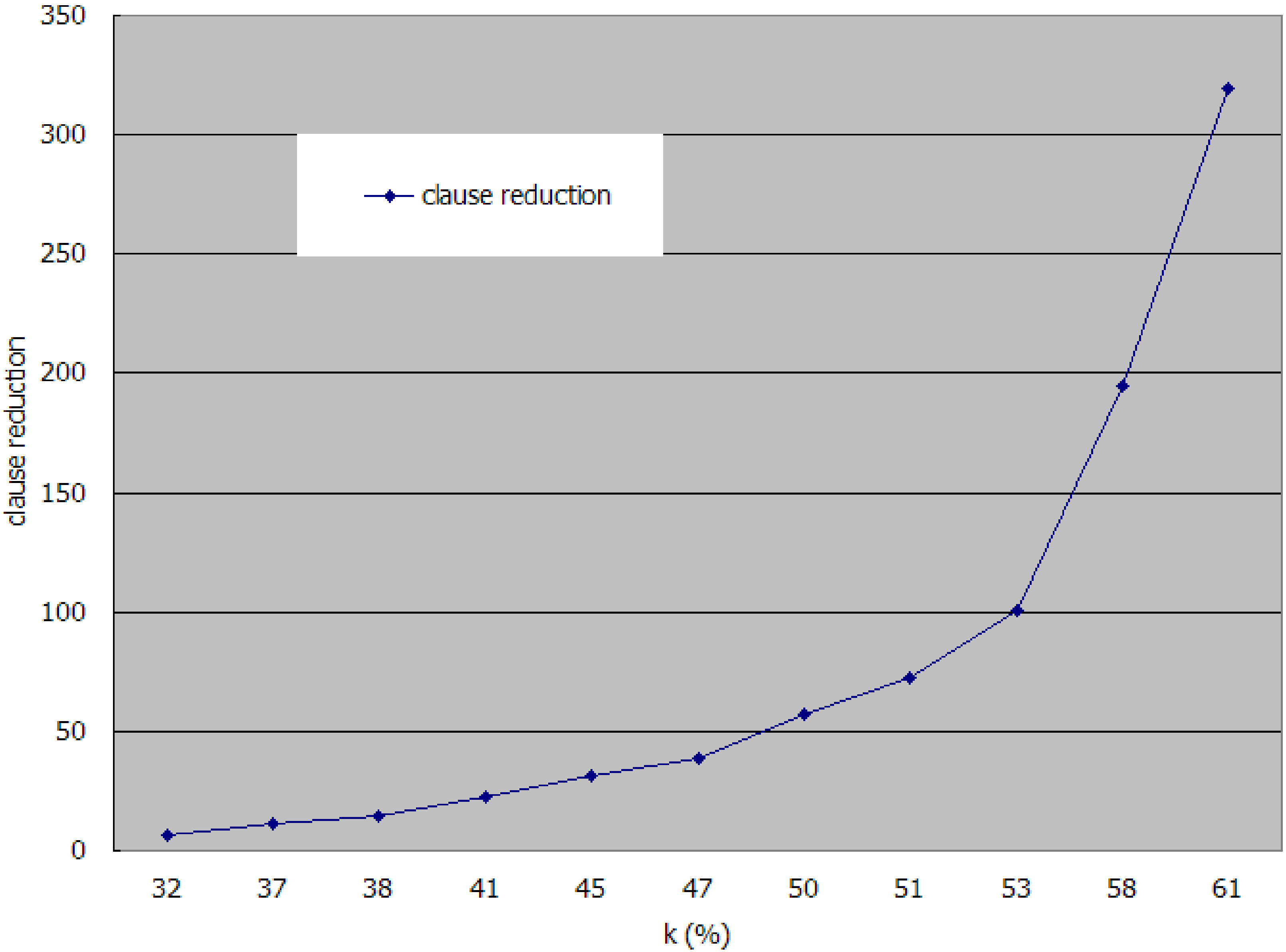
Time Reduction



Variable Reduction Ratio



Clause Reduction Ratio



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 greately reduced

+ Number of clauses is greately reduced

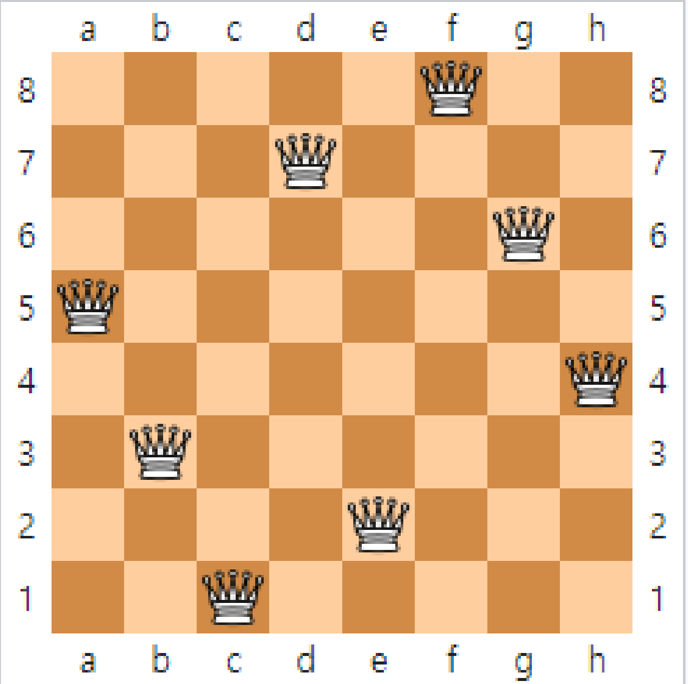
+ Execution time is greately reduced

+ Finally, encoding time is greately reduced

Thank You!!

Another Example: 8 Queen Puzzle

- Placing 8 chess queens on an 8×8 chessboard so that no two queens threaten each other
  - Thus, a solution requires that no two queens share the same row, column, or diagonal.
  - There are 92 solutions.
- The asymptotic growth rate of the number of solutions for n queen puzzle is (0.143 n)n.
  - the exact number of solutions for n queen puzzle is only known for n ≤ 27,



The only symmetrical solution to the eight queens puzzle (up to rotation and reflection)