Investigating the Effects of Extended Encoding Against Minimal Encoding

Evan Wilde, Sebastien Guillemot, Shayla Redlin, Laura Grondahl
Department of Computer Science, University of Victoria, Canada.

***Abstract*—The Sudoku game was popular in Japan since the mid-80's, but in 2005 had an explosion in western countries. The popular game can be encoded as a SAT problem and given to a SAT solver to get a solution to the puzzle. Various encodings of the Sudoku problem have been presented. In this paper we explore the scalability of extended encoding and minimal encoding of Sudoku as SAT over puzzle difficulty levels and board sizes.**

***Keywords*—Sudoku, SAT-solver, minimal encoding, extended encoding**

I. Introduction

The game of Sudoku became popular in Japan in the mid-80's [2], but in 2005 had an explosion in western countries. The popular game can be encoded as a SAT problem and given to a SAT solver to get a solution to the puzzle. Various encodings of the Sudoku problem have been presented. In this paper we explore the scalability of extended encoding and minimal encoding of Sudoku as SAT over puzzle difficulty levels and board sizes. We ran the encodings using the "minisat"[1] SAT solver to get our results. All of our results are available on a GitHub repository[2].

II. Related Work

There has been research in the area of finding cpu and memory efficient encodings of sudoku as a SAT problem [1,2]. Both papers look at a way to get better performance from a SAT solver on the sudoku problem. *Sudoku as a SAT Problem* simply presents an extended encoding which generates more rules for the board, which limit the number of ambiguities that the SAT solver must attempt to solve, thus increasing the CPU performance. The full extended encoding is a $O(n^4)$ algoroithm, so as the board size grows, the feasibility of generating the necessary clauses drops. *Optimized CNF Encoding for Sudoku Puzzles* looks at obtaining good results from the SAT solver, but also good results from the encoding process itself [1], decreasing the cpu time, memory requirements,and the number of clauses to be generated. Our paper does not look into the efficient encoding, only extended and minimal encodings.

III. Methodologies

To test the benefits and drawbacks of the presented minimal and extended encodings, we performed the following tests to determine how each encoding scales across the difficulty of the puzzle and the size of the puzzle. The puzzles we used for testing purposes are available in the GitHub repository.

1) Minimal encoding on 9×9 puzzles of "easy", "medium", and "Lex Luthor" difficulty levels.
2) Minimal encoding on unsolvable 9×9 puzzles of "easy", "medium" and "Lex Luthor" difficult levels.
3) Extended encoding on 9×9 puzzles of "easy", "medium", and "Lex Luthor" difficulty levels.
4) Extended encoding on unsolvable 9×9 puzzles of "easy", "medium" and "Lex Luthor" difficult levels.
5) Minimal encoding on 16×16 puzzles of "easy", "medium", and "Lex Luthor" difficulty levels.
6) Extended encoding on 16×16 puzzles of "easy", "medium", and "Lex Luthor" difficulty levels.

We sample the results on a 3rd Gen. intel core i5-3337U cpu clocked at 1.80Ghz with 4Gb of RAM.

A. Data sample

We are using an online board generator[3] with difficulties ranging from "easy" to "Lex Luthor". Easier puzzles have many blocks originally filled in, whereas puzzles at "Lex Luthor" difficulty only have a few positions filled in, adding more potentially ambiguous cases. On a 9×9 board, an "easy" board may have more than 50% of the squares filled in, whereas a "Lex Luthor" board may have as few as 4 squares filled in, which leads to more ambiguous cases and potentially many solutions for a given puzzle.

IV. Results

This section contains the results of the experiment. The results are generated on a 3rd Gen. intel core i5-3337U cpu clocked at 1.80Ghz with 4Gb of RAM using "minisat". The resulting times only include the reported time taken for the SAT solver to run, not the time for the encoder to run on each puzzle.

[1] http://minisat.se/MiniSat.html
[2] https://github.com/etcwilde/sudokusolver
[3] http://sudoku-puzzles.merschat.com/

A. Test 1: Minimal encoding on 9×9 *puzzles of "easy", "medium", and "Lex Luthor" difficulty levels.*

Difficulty	Variables	Clauses	Parse time (s)
Easy	729	1895	0.00
Medium	729	1926	0.00
Lex Luthor	729	7246	0.00

TABLE I: Minimal 9×9 Parse Information

Difficulty	Memory	Time (s)	Decisions	Conflicts	Propagations
Easy	19.00	0	1	0	729
Medium	19.00	0	5	0	729
Lex Luthor	19.00	0.003333	95	4	851

TABLE II: Minimal 9×9 Processing Information

B. Test 2: Minimal encoding on unsolvable 9×9 *puzzles of "easy", "medium" and "Lex Luthor" difficult levels.*

Difficulty	Variables	Clauses	Parse time (s)
Easy	729	1863	0.00
Medium	729	1813	0.00
Lex Luthor	729	6864	0.00

TABLE III: Unsolvable Minimal 9×9 Parse Information

Difficulty	Memory	Time (s)	Decisions	Conflicts	Propagations
Easy	19.00	0.003333	0	0	646
Medium	19.00	0.003333	0	0	512
Lex Luthor	19.00	0	86	53	2518

TABLE IV: Unsolvable Minimal 9×9 Processing Information

C. Test 3: Extended encoding on 9×9 *puzzles of "easy", "medium", and "Lex Luthor" difficulty levels.*

Difficulty	Variables	Clauses	Parse time (s)
Easy	729	1895	0.00
Medium	729	1953	0.00
Lex Luthor	729	9217	0.00

TABLE V: Extended 9×9 Parse Information

Difficulty	Memory	Time (s)	Decisions	Conflicts	Propagations
Easy	19.00	0	1	0	729
Medium	19.00	0	2	0	729
Lex Luthor	19.00	0	64	0	729

TABLE VI: Extended 9×9 Processing Information

D. Test 4: Extended encoding on unsolvable 9×9 *puzzles of "easy", "medium" and "Lex Luthor" difficult levels.*

Difficulty	Variables	Clauses	Parse time (s)
Easy	729	1863	0.00
Medium	729	1813	0.00
Lex Luthor	729	8429	0.00

TABLE VII: Unsolvable Extended 9×9 Parse Information

Difficulty	Memory	Time (s)	Decisions	Conflicts	Propagations
Easy	19.00	0	0	0	646
Medium	19.00	0.003333	0	0	512
Lex Luthor	19.00	0.003333	0	0	252

TABLE VIII: Unsolvable Extended 9×9 Processing Information

E. Test 5: Minimal encoding on 16×16 *puzzles of "easy", "medium", and "Lex Luthor" difficulty levels.*

Difficulty	Variables	Clauses	Parse time (s)
Easy	4096	8716	0.02
Medium	4096	30977	0.02
Lex Luthor	4096	73448	0.02

TABLE IX: Minimal 16×16 Parse Information

Difficulty	Memory	Time (s)	Decisions	Conflicts	Propagations
Easy	20.00	0.016666	0	0	1367
Medium	20.00	0.026666	267	73	15880
Lex Luthor	21.00	0.036666	2446	272	27704

TABLE X: Minimal 16×16 Processing Information

F. Test 6: Extended encoding on 16×16 *puzzles of "easy", "medium", and "Lex Luthor" difficulty levels.*

Difficulty	Variables	Clauses	Parse time (s)
Easy	4096	8761	0.02
Medium	4096	32205	0.03
Lex Luthor	4096	91595	0.04

TABLE XI: Extended 16×16 Parse Information

Difficulty	Memory	Time (s)	Decisions	Conflicts	Propagations
Easy	20.00	0.019999	0	0	1367
Medium	20.00	0.033333	13	3	4214
Lex Luthor	23.00	0.029999	444	4	4371

TABLE XII: Extended 16×16 Processing Information

V. Discussion

Our original hypothesis was that the minimal encoding would take less cpu time; however, in all test cases, the number of propagations, decisions, and cpu time decreased from the minimal encoding to the extended encoding.

		Minimal Encoding				Extended Encoding			
size	level	vars	clauses	dec	prop	vars	clauses	dec	prop
9×9	easy	729	1895	1	729	729	1895	1	729
9×9	medium	729	1926	5	729	729	1953	2	729
9×9	lex	729	7246	95	851	729	9217	64	729
16×16	easy	4096	8716	0	1367	4096	8761	0	1367
16×16	medium	4096	30977	267	15880	4096	32205	13	4214
16×16	lex	4096	73448	2446	27704	4096	91595	444	4371

TABLE XIII: Minimal encoding versus Extended encoding on solvable puzzles

		Minimal Encoding				Extended Encoding			
size	level	vars	clauses	dec	prop	vars	clauses	dec	prop
9×9	easy	729	1863	0	646	729	1863	0	646
9×9	medium	729	1813	0	512	729	1813	0	512
9×9	lex	729	6864	86	2518	729	8429	0	252

TABLE XIV: Minimal encoding versus Extended encoding on unsolvable puzzles

In table XIII, we can see that the growth of the decisions and propagations over difficulty levels is less given an extended encoding of the board over a minimal encoding of the board. This trend holds true as the size of the puzzle grows.

In table XIV, we can see that both the extended encoding and the minimal encoding ran in the same time for unsolvable puzzles where there were no possible ambiguities. The "Lex Luthor" puzzle does not necessarily have a single solution so we see deviation between the two methods. In this case, we can see that the extended encoding was able to arrive at the conclusion that the board was unsatisfiable in fewer decisions and propagations.

A. Limitations

Theoretically our program can use any $n \times n$ puzzle, so long a n is a perfect square; however, due to the limitations of encoding numbers larger than 35, our program can only take puzzles up to 25×25. A possible method of extending the usage of the solver is to allow a delimiter between values in each cell.

VI. Future Work

As stated [1], the growth of the extended encoding is $O(n^4)$, which may cause the total runtime of the extended encoding to be larger than the minimal encoding simply due to the encoding process. It would be interesting to further the study on 81×81 puzzles and 100×100 puzzles to see at what point the extended encoding is no-longer feasible due to the encoding runtime. Furthermore, it would be interesting to investigate the efficient encoding and compare that against the extended and minimal encodings.

VII. Threats to Validity

In the paper presented by Lynce it states that the extended encoding requires two extra properties: that the puzzles only have one solution and that puzzles can be solved with only reasoning. The boards we were testing with, specifically the "Lex Luther" difficulty are not guaranteed to maintain these extra properties. From visual inspection, the minimal and extended encodings were able to correctly generate two correct solutions for the same puzzle, but decreasing the accuracy of the comparison.

Our original hypothesis was using cpu time as a measurement of difficulty , but we found that the number of propagations and decisions was a better measurement of the difficulty of a given puzzle.

VIII. Conclusion

We found that extended encoding ran with fewer propagations, decisions, and cpu time than minimal encoding, but with higher memory usage and more clauses. The growth of the clauses may limit the size of boards that the extended encoding scheme is feasible for encoding.

Our program is able to run on $n \times n$ puzzles up to $n = 25$ where n is a perfect square.

References

[1] Gihwon Kwon and Himanshu Jain. Optimized CNF Encoding for Sudoku Puzzles. *Engineering*, pages 1–5, 2006.

[2] I Lynce and J Ouaknine. Sudoku as a SAT Problem. *Symposium A Quarterly Journal In Modern Foreign Literatures*, pages 1–9, 2006.