Knowledge Representation:
The influence of givens in Sudoku SAT-solving

4 October 2019

1 Introduction

A Sudoku is a logic-based number-placement puzzle. In each row, column, and 3 by 3 sub squares an integer from 1 to 9 must placed once and only once. This means that the sum of the numbers in each row, column, or sub square is equal[3]. A Sudoku is presented with some known numbers and some black entries. The numbers that are given can't be moved or changed at any time. Because every Sudoku has a unique solution, a Sudoku can be solved with 'reasoning'. So a Sudoku is a NP-complete problem and can be encoded as a SAT problem[2].

2 Design SAT-solver

2.1 Davis-Putnam algorithm

The Davis-Putnam algorithm consists of two parts, the simplification and the picking of a truth value for an unassigned element. The simplification uses 3 checks. The first one is checking for a tautology, both polarities of a literal in the same clause. This is always true. The second one is checking for a pure literal, this means a literal occurs in all clauses solely in or it's positive or it's negative polarity. This literal can made both True or False. The last check is to see if there is a unit clause, a clause that only contains one element. If this is the case, this element is set to True.[1]

The second part of the algorithm is the picking of a truth value for an unassigned element is also called a split. In the basic Davis-Putnam this is done by picking a random element. However, there a several heuristics available for picking an element to assign a truth value to. Some of which will be touched upon later. If the list of clauses can't be simplified to an empty list, it is satisfiable. If not backtracking must occur and a new option can be tried, or if all options didn't yield a result, the set in unsatisfiable. [1]

2.2 Implementation

The algorithm is implemented as follows: First there is a check for a tautology. This only happens once. After the clauses of the clausal normal form are reduced based on the truth values of the given elements. This continues on by checking for unit clauses and pure literals. Whe a pure literal is found, this will always be set to True. Even if it might entail the negative polarity of the literal. After these checks the clauses are reduced again. This continues on until the clauses can't get reduced further. After which a random clause is made true. This continues on until a result is found or an empty clause is found. In the case of the latter, the polarity of the last randomly selected literal is reversed. If this again leads to an empty clause, the algorithm backtracks a level up.

It is important to note that the random selection of elements in the implementations is not completely unbiased. In the implementation a random clause of the list of clauses in clausal normal form is selected, and in that clause a random element is selected. This means that technically a more occurring element is more likely to be selected. However, it is assumed this is negligible.

3 Heuristics

3.1 Dynamic Largest Individual Sum

The Dynamic Largest Individual Sum is a simple count heuristic for the Davis-Putnam algorithm. The occurrence of each literal in the set of clauses is counted and the most common is given a truth value. The different polarities of clauses are considered separately. In the formulation of the heuristic the truth value is given based on a check. If the occurrence of positive literal is higher than or equal to the occurrence of the negative literal, the literal is made True, otherwise the literal is made False.[4]

The implementation of this heuristic in this paper does not utilise the check, because the sorting algorithm used should always provide the literal the highest occurrence, so it can not be lower than the other polarity if the same literal.

The choice was made to implement this specific heuristic, because it seems logical to take the most occurring literal and make it True. This way the maximal amount of clauses will be resolved. Which suggest a shorter solving process. Even if it is found that the polarity should be False, the maximal amount of elements get removed, which seems to increase the chance of a unit clause occurring.

3.2 Two-sided Jeroslow-Wang

The Jeroslow-Wang heuristics were proposed by Jeroslow and Wang. These heuristics are slightly more sophisticated than the Dynamic Largest Individual Sum, because the clause length is taken into account. The smaller the clause, the higher the value it is given. There are two versions of the Jeroslow-Wang, in this case the two-sided version was used, which considers the polarities together.

For each literal per occurrence 2 to the power minus the absolute clause length is calculated, and this is summed over all occurrences. The literal with the highest result of this calculation is selected. Now the different polarities come into play and if the positive polarity results to a higher or equal calculated value as the negative polarity, the positive one is made True, also the Negative one is made True. [4]

This heuristic was implemented, because taking account the length of a clause seems to make sense. When a clause is small, there are less options to make elements True. So to make not end up with empty clauses it is important that at least one element of those limited options is True.

4 Hypothesis

To make a unique Sudoku you need at least 17 clues and not more than 77 clues [5]. It looks harder to solve, for example, a Sudoku with 17 clues than a Sudoku with 18 or more clues. So the hypotheses of this article is: Sudoku's with fewer given elements need more splits and backtracks that Sudoku's with more given elements.

5 Experimental design

In order to test the hypotheses we used 372 different 9 by 9 Sudoku's with different amount of clues, between 18-25. It is important to note that these aren't evenly distributed. The Sudoku's were sorted by the number of clues and for each number of clues, the average number of splits, random or heuristic based truth value assignments, the number of polarity switches, and backtracks, back to an upper level, were calculated. A focus is placed on the splits and the backtracks for the basic SAT-solver, the Dynamic Largest Individual Sum SAT-solver (DLIS), the two-sided Jeroslow-Wang SAT-solver and the average of all three. The correlation is tested by the Pearson correlation coefficient, because a linear correlation is expected between the backtracks or the number of uncertain truth assignments with the number of clues and the variables all were measured on an interval scale.

6 Experimental Results

The Pearson-correlation showed that there is no-significant, but a strong relation between the backtracks and the amount of given elements ($r = -0{,}81$; $p = 0.09$). There is also no-significant, but strong relation between the selected clauses and the amount of given elements ($r = -0{,}82$; $p \leq 0.05$). (See Figure 1 and table 1) Overall the two-sided Jeroslow-Wang heuristic got the best correlation, there is significant . The backtracks correlated with -0,82 ($p \leq 0.05$) with the amount given elements and the selected clauses correlated with -0.95 ($p \leq 0.01$). The lowest correlation is found at the Dynamic Largest Individual Sum heuristic

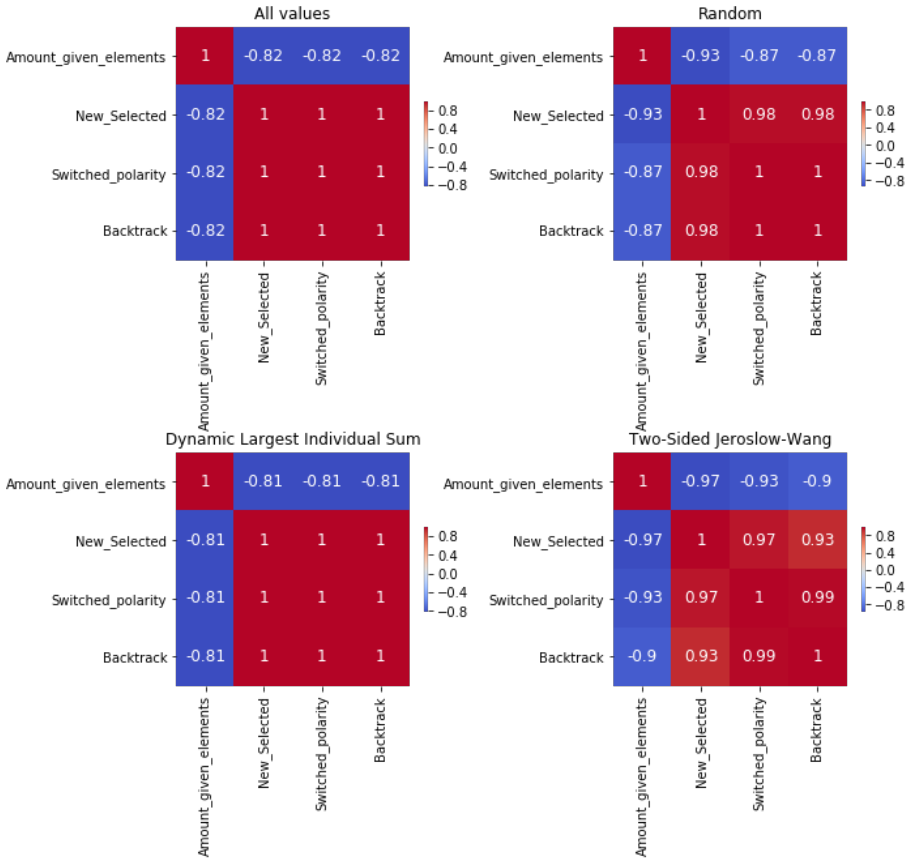

Figure 1: Correlation matrices of the different heuristics

with a Pearson-correlation of -0,81 ($p > 0.05$) for both the selected clauses and the backtrack.

7 Conclusion

There is a strong non significance correlation between the amount of given elements and the new selected clauses and the backtracks. This means the hypothesis is rejected. It is important to note that the sample size was small, only 372 Sudoku's and the spread of the amount of givens wasn't large. Furthermore, one of the chosen heuristics, the Dynamic Largest Individual Sum, unexpectedly under-preformed greatly. This influenced the averaged result. If further research is done, it would benefit from a bigger spread of givens, a bigger spread of heuristics, and a bigger sample set.

Table 1: The P-values for the calculated correlation.

	All	**Random**	**DLIS**	**2-sided JW**
New_selected	0.089736	0.020321	0.093104	0.005507
Backtrack	0.092274	0.054015	0.095109	0.038639

Table 1: The P-values for the correlation between the given elements and the times a new truth value was selected or a backtrack to an upper level was preformed per heuristic

References

[1] Carla P Gomes, Henry Kautz, Ashish Sabharwal, and Bart Selman. Satisfiability solvers. 2007.

[2] Ines Lynce Ist, Inês Lynce, and Joël Ouaknine. Sudoku as a sat problem. In *Proceedings of the International Symposium on Artificial Intelligence and Mathematics (AIMATH)*, pages 1–9, 2006.

[3] Timo Mantere and Janne Koljonen. Solving, rating and generating sudoku puzzles with ga. In *2007 IEEE congress on evolutionary computation*, pages 1382–1389. IEEE, 2007.

[4] Joao Marques-Silva. The impact of branching heuristics in propositional satisfiability algorithms. In *Portuguese Conference on Artificial Intelligence*, pages 62–74. Springer, 1999.

[5] Gary McGuire, B Tugemanny, and G Civarioz. There is no 16-clue sudoku: Solving the sudoku minimum number of clues problem via hitting set enumeration. *arXiv preprint arXiv:1201.0749*, 2013.