Sliding Tile Puzzles

Robert Clausecker and Benjamin Kaiser
Zuse Institute Berlin
Berlin, Germany
{clausecker,kaiser}@zib.de

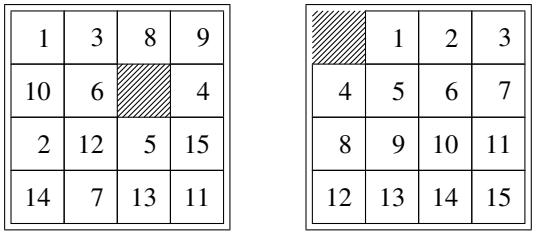

Fig. 1. a permuted and a solved *15 puzzle*

***Abstract*—We translate 5×5 sliding tile puzzle (24 puzzle) instances into CNF formulæ to compare the performance of SAT solvers with standard heuristic search methods in this domain. We find that SAT solvers still have a long way to go before they might be competitive in this problem domain.**

I. Introduction

Sliding tile puzzles comprise an $n \times m$ rectangular tray holding square tiles numbered 1 to $(nm - 1)$ with one spot empty. The objective is to permute the tiles such that their are arranged in order. To get there, the state of the puzzle can be changed by shifting tiles adjacent to the blank spot into the blank spot.

In the domain of heuristic search, sliding tile puzzles are frequently used as NP-hard model problems for graph search methods. They are particularly useful due to their simple and regular structure and admit use of many advanced search techniques.

With this submission, we would like to understand how well SAT methods might be suitable for solving this problem. While SAT solvers lack dowmain specific heuristics, they are able to attack the problem in ways that aren't really accessible to tree-search methods, e.g. by drawing conclusions from parts of the puzzle configuration that can be reused in other parts.

II. Encoding

The basic decision problem is: can a given configuration of the $(n \times n - 1)$ puzzle be transitioned into the solved configuration within k moves? This decision problem is encoded into a SAT instance by creating $k + 1$ sets of variables, each representing one puzzle configuration. Clauses are added encoding that adjacent configurations must be related by performing a single move. Furthermore, the first configuration must be equal to the problem configuration and there must exist a configuration equal to the solved configuration.

Each configuration is represented as an array of $n \times n$ vectors of literals where each vector $t_{i,j}$ represents the number of the tile at grid location (i, j). The grid is rotated horizontally and vertically such that the blank spot (numbered 0) is always in the top left corner. Two vectors of bits h and v encode in one-hot encoding where the top left tile ends up after the grid is rotated.

The move relation is encoded using two literals m_0 and m_1 encoding if the move taken was up, down, left, or right. By checking the polarity of these literals in the model found by the solver, the solution to the puzzle can be extracted.

Using these literals, we then check if the tiles on the board moved according to the move taken. We also check h and v to ensure that moves across the border do not occur.

III. Submitted Benchmarks

As a sample instance, we picked problem 50 of Korf's instances of the 24 puzzle [1]. As the full problem with its 113 step solution is too difficult to be solved by current SAT solvers, we simplified it by tracing the solution of the problem and taking the puzzle configurations obtained with distances $k = 30 \ldots 60$ to the solved configuration. With rising k, the configurations become progressively harder to solve.

For each such k, two SAT instances were generated. One instance has a move budget of k and is satisfiable. The other instance has a move budget of $k - 2$ and is unsatisfiable. This way, both the capability to solve SAT and UNSAT instances is exercised using the same type of instance.

Acknowledgment

We want to express our gratitude towards the organisers of the SAT Competition 2021 for making such an event possible. Additionally we like to thank Florian Schintke for his support and the IT and Data Services members of the Zuse Institute Berlin for providing the infrastructure.

References

[1] Richard E. Korf and Ariel Felner, "Disjoint Pattern Database Heuristics", Artificial Intelligence 134(1–2), p. 9–22, 2020.