Fill-a-Pix Puzzle as a SAT Problem

Aye Myint Myat, Khine Khine Htwe, Nobuo Funabiki
University of Information Technology, Yangon, Myanmar
ayemyintmyat@uit.edu.mm, khinekhine@uit.edu.mm, funabiki@okayama-u.ac.jp

Abstract

Fill-a-Pix Puzzle is a Picture Logic Puzzle that has not been solved as a SAT Problem as well as there is no SAT Conjunctive Normal Form (CNF) Encoding Method to solve this puzzle yet. There are several practical SAT problems in various fields such as Artificial Intelligence (AI), Automatic Theorem Proving, Circuit Design, etc. Fill-a-Pix puzzle is also one of the SAT problems. This research proposes the SAT CNF Encoding Method to solve Fill-a-Pix Puzzle as a SAT Problem using SAT Solvers. The proposed SAT CNF Encoding Method will be executed on different standard SAT solvers – MiniSAT, CryptoMiniSAT and RSAT. The evaluation is presented regarding the CPU Execution Times of each solver for executing the proposed SAT CNF Encoding, the Number of Variables and Clauses produced by the proposed SAT CNF Encoding as well as the Comparison of Fill-a-Pix Puzzle with the other Similar Puzzles such as Sudoku and Slitherlink based on the Number of Variables and Clauses produced by the proposed SAT CNF Encoding when executing Puzzle Sizes above 50 x 50.

Key Words- Fill-a-Pix Puzzle, SAT CNF Encoding, MiniSAT, CryptoMiniSAT, RSAT

1. Introduction

Satisfiability SAT problem is the first problem proved to be NP-Complete according to Cook-Levin theorem. Several problem instances with a lot of variables and formulas with many symbols can be solved by using SAT algorithms which can also be used for solving SAT problems. Several logic problems and puzzles are trying to be solved as SAT problems using SAT solvers. Sudoku is one of the most basic and popular puzzles solving as a SAT problem [1, 2, 3, 4]. Not only Sudoku puzzles are solving as a SAT problem but also the other puzzles such as Killer Sudoku [6], Binary Puzzle [7] and Slitherlink [8], Edge Matching Puzzles [9] are now solving as a SAT problem. There are also several researches in finding optimized CNF encoding for them [5]. Solving logic puzzles as SAT problems could also get advantages in solving other problems like coloring problems, mathematical problems and also in real word applications such as data hiding techniques like steganography, systematic problem solving. However, many logic puzzles are still under researches for solving as SAT problems. Fill-a-Pix puzzle is one of those which could be solved as a SAT problem. Even though, many researches have been conducted for Sudoku puzzle, research on Fill-a-Pix puzzle as a SAT problem still needs to be carried out.

Hence, in this research paper, the SAT CNF encoding to solve Fill-a-Pix puzzle as a SAT problem is proposed and the proposed SAT CNF encoding is evaluated on different standard SAT solvers and finally, conclude with suggesting the possible further improvements.

2. Related Work

Artificial intelligence in the game field became more important and widely used. Currently, computers are applied in many areas of game fields and logic puzzles are also being solved using computers to surpass the humans. Puzzle solving can be used in Artificial Intelligence (AI) as well as in the important mathematical problems such as coloring problems. In additions, puzzle solving could provide a better outlook in approaching real world tasks.

This work starts from the idea of solving Sudoku puzzles as a SAT problem [2, 3, 4] which leads to apply Fill-a-Pix puzzle as a SAT problem. There are several research papers in solving Sudoku puzzle as a SAT problem and optimizing SAT encoding [5]. The work in this paper presents the applying of SAT CNF encoding in solving Fill-a-Pix puzzles. Afterwards, different SAT solvers are used to evaluate the proposed SAT CNF encoding for solving Fill-a-Pix puzzles.

3. Background Theory

3.1. Logic Puzzles

A logic puzzle is a puzzle which is a kind of game or problem that tests a person's knowledge or ingenuity and it is derived from the mathematics field of deduction. They are mainly categorized into two types 1) Picture Logic Puzzles and 2) Number Logic Puzzles. Pic-a-Pix, Sym-a-Pix, Link-a-Pix, Fill-a-Pix, Maze-a-Pix, Dot-a-Pix, Cross-a-Pix, Block-a-Pix are types of Picture Logic Puzzles. After solving the logic puzzles, the hidden pictures would be usually discovered from the puzzles and so, they are called Picture Logic Puzzles. On the other hands, Sudoku, Kakuro, Battle Ships, Hitori, Slitherlink, Ha Shi, Calcu Doku, Nurikabe, Skyscrapers and Tic-Tac-Logic are usually kinds of Number Logic Puzzles whereas the player

needs to work with numbers to solve the puzzles. In this research paper, Fill-a-Pix puzzle, one of the Picture Logic Puzzle, is applied as a SAT problem.

3.2. Fill-a-Pix Puzzle

Fill-a-Pix puzzle is a logic puzzle given a grid made up of cells, containing various numbers between 0 and 9 in some cells [11]. The player has to solve the puzzle by coloring the neighbor cells with black according to the given number in a cell and the player tries to find the final hidden picture. There is only one unique solution for each Fill-a-Pix puzzle as well as each puzzle is solved only with reasoning and does not require to use guessing.

Fill-a-Pix puzzles are invented by Trevor Truran and developed by Conceptis. These puzzles offer the ultimate mix of logic, art and fun. Fill-a-Pix puzzle is a Minesweeper-like puzzle based on a grid with a pixel-art picture hidden inside. Using logic alone, the player determines which squares are painted and which should remain empty until the hidden picture is completely exposed. The rules of Fill-a-Pix puzzles are very simple and easy to learn. Fill-a-Pix puzzles come in black and white. They are available in different sizes as well as difficulty levels. They would require from five minutes to several hours to solve. There are two levels in Fill-a-Pix puzzles as described as follows:

1) **Basic Logic**: Requires analysis of one clue at a time and are straightforward to solve. Figure 1 shows the example of Basic Logic of Fill-a-Pix puzzle of size 5 x 5. Figure 1 (a) shows the unsolved puzzle and Figure 1 (b) shows the final solved puzzle revealing the discovered hidden picture of "House".

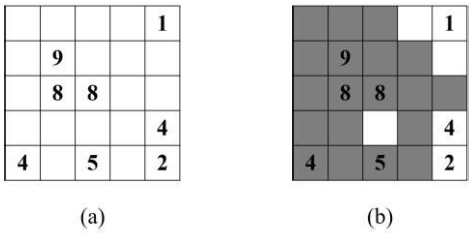

Figure 1 (a). Basic Logic Fill-a-Pix Puzzle
(b). Solution for **Hidden Picture of "House"**

2) **Advanced Logic**: Contains additional situations where two clues simultaneously affect each other as well as the squares around them making these puzzles very challenging and rewarding to solve. Figure 2 shows the example of Advanced Logic of Fill-a-Pix puzzle of size 5 x 5. Figure 2 (a) shows the unsolved puzzle and Figure 2 (b) shows the final solved puzzle revealing the discovered hidden picture of "Boy".

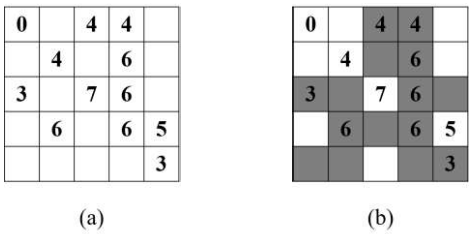

Figure 2 (a). Advanced Logic Fill-a-Pix Puzzle
(b). Solution for **Hidden Picture of "Boy"**

3.3. Satisfiability SAT

Satisfiability SAT problem determines whether a given Boolean formula is satisfiable or unsatisfiable [10]. It determines the formula is satisfiability (true) if there exists an interpretation or unsatisfiable (false) if there is no assignment for the given Boolean formula. The propositional logic formula or Boolean expression is constructed with variables, operators (Conjunction AND/$\wedge$, Disjunction OR/$\vee$ and NOT negation/$\neg$) and parentheses. After that, the appropriate values True or False will be assigned to these variables. In this case, SAT checks that the given formula is said to be satisfiable if the assignment makes true, otherwise, unsatisfiable. There are two variables or literals in a SAT problem called positive literal or negative literal. In SAT, the clauses of the formula are in Conjunctive Normal Form (CNF) which is explained in the later section. **Definition 3.1** shows the SAT problem.

Definition 3.1. (SAT Problem) SAT Problem is a decision problem which checks whether a given formula is satisfiable or not. The following is the example of SAT Problem.

Conjunctive Normal Form of the given formula of SAT is described as follow.

$$(x_1 \vee \neg x_2) \wedge (\neg x_1 \vee x_2 \vee x_3)$$

By assigning appropriate True or False values to each variable, the formula is determined satisfiable or not.

Let,

$x_1 = False$

$x_2 = False$

$x_3 = False$

By assigning the values into the formula,

$$(False \vee \neg False) \wedge (\neg False \vee False \vee False)$$

After solving the formula, the final result is evaluated to be true, in other words, the given formula is satisfiable.

$$(False \vee \neg False) \wedge (\neg False \vee False \vee False)$$
$$(False \vee True) \wedge (True \vee False \vee False)$$
$$(True) \wedge (True)$$
$$True$$

The first problem proved to be NP-Complete is SAT problem according to Cook-Levin theorem. There are problem instances that have thousands of variables and

formulas with millions of symbols. Those instances can be solved by using SAT algorithms. These SAT algorithms can also be used to solve SAT problems from Artificial Intelligence (AI), Automatic Theorem Proving, Circuit Design, etc.

3.4. Conjunctive Normal Form (CNF)

CNF is the abbreviation of Conjunctive Normal Form or Clausal Normal Form in which one or more clauses are in conjunction and literals of the clauses are in disjunction. **Definition 3.2** shows the example of CNF.

Definition 3.2. (CNF) CNF is an AND of ORs whereas the disjunction ($\vee$) of literals and the conjunction ($\wedge$) of clauses.

$$(x_1 \vee \neg x_2) \wedge (\neg x_1 \vee x_2 \vee x_3)$$

3.5. SAT Solver

Modern SAT solvers have greatly influenced on software verification, constraint solving in artificial intelligence as well as operations research. There are several SAT solvers that could solve SAT CNF encoding. SAT solvers that will be used in this research paper for the evaluation of the proposed SAT CNF encoding are 1) MiniSAT 2) CryptoMiniSAT and 3) RSAT.

3.5.1. MiniSAT

MiniSAT is a minimalistic, open-source SAT solver, developed to help researchers and developers alike to get started on SAT [12]. It is an Incremental SAT Solver and uses the features of DPLL, VSIDS, 2 Watched Literal and Conflict-Driven Backjumping.

Its main impressed feature is the Conflict Minimization using Resolution technique. It starts from the ordinary conflict clause and analyzes the conflict called Conflict Analysis. Then, it drives Backjumping technique which made going up more levels called Safe Jump. This would reduce the search space and increases the efficiency. Finally, it applies Subsumption Resolution greedily and minimizes the conflicts called Conflict Minimization.

3.5.2. CryptoMiniSAT

CryptoMiniSAT is a modern, multi-threaded, feature-rich, simplifying SAT solver [13]. It is a SAT solver for Cryptographic Problems and to optimize a SAT solver for cryptographic problems, the solver's input language is extended to support XOR operation that is common in cryptography. It could recover XORs from a CNF. It uses XOR Recovery and Manipulation technique in which, it could be tried to XOR the XORs together to obtain new information.

3.5.3. RSAT

RSAT is a complete Boolean satisfiability solver [14]. RSAT uses the phase selection heuristic that is oriented toward reducing work repetition and a frequent restart policy. The features used in RSAT are DPLL, Resolution, 2 Watched Literal and Phase Selection Heuristic.

It is an SAT solver based on DPLL framework together with using Backtracking. The main feature of RSAT is its heuristic algorithm in which the algorithm frequently restarts that would decrease performance on some SAT instances. The goal is to cache partial solutions to subsets of the formula and reuse them after a restart by using the idea of remembering last assignment of each variable and using it first in branching. As a result, the phase saving stabilizes positive effect of restarts and the best results in combination with non-chronological backtracking.

4. Proposed System

The proposed system design is illustrated in Figure 3. The steps for the whole system are as follows:

1) Dataset would be obtained from Fill-a-Pix puzzle official website.
2) Based on the rule of playing Fill-a-Pix puzzle, the constraints will be proposed, transformed into CNF and encoded them to be solved in SAT solver (The proposed constraints and SAT CNF encoding are described in Section 6).
3) After solving using SAT solver, the resulted SAT solution is decoded and displayed the puzzle solution.
4) When solving SAT CNF encoding, three SAT solvers are used to evaluated the proposed SAT CNF encoding: MiniSAT, CryptoMiniSAT and RSAT.

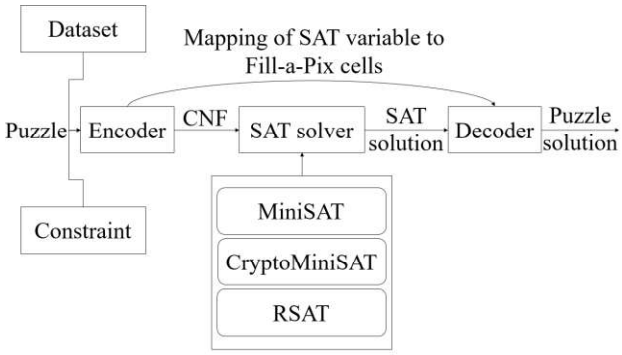

Figure 3. Proposed system design

5. Problem Definition

Input

- n logical (binary) variables: $x_1, \ldots, x_n$
  - Literal: x_i, $\neg x_i$ (Negation)
- m clauses: $C_1, \ldots, C_m$

Output

- True – False (0 – 1) assignment to the variables

Constraint

- The number in a box is the number of black boxes among adjacent to it or itself.

Objective/Cost Function

- To solve a given Fill-a-Pix puzzle as a SAT problem.

6. SAT CNF Encoding

In this section, the proposed SAT CNF Encoding for Fill-a-Pix puzzle will be presented.

First and foremost, the number of variables for a puzzle will be considered. There will be two types of variables

1) Cell Variables: Represents each cell. x_{0ij}, where cell x at row i and column j. (0 is inserted at the front to distinguish Cell Variables and Number Variables). **Definition 6.1.** describes how to calculate the Cell Variables of a given puzzle.
2) Number Variables: Represents the given number in each cell. In Fill-a-Pix puzzle, each cell has 10 numbers from 0 to 9 (10 variables). x_{ijk}, where cell x at row i and column j which has number k. **Definition 6.2.** describes how to calculate the Number Variables of a given puzzle.

Definition 6.1. (Cell Variable) The number of Cell Variable is the product of row and column of a given puzzle.

$$Number\ of\ Cell\ Variables = Row \times Column$$

Definition 6.2. (Number Variable) The number of Number Variable is the product of row and column and the possible numbers in a cell of a given puzzle.

$$Number\ of\ Number\ Variables = Row \times Column \times Number$$

Definition 6.3. (Total Variable) The total number of variables to be used for a given puzzle is the sum of the number of Cell Variables and the number of Number variables.

$$Total\ Number\ of\ Variable = Number\ of\ Cell\ Variables + Number\ of\ Number\ Variables$$

The given puzzle information and the constraints will be encoded into the corresponding variables according to the above definitions. By using the constraint for the puzzle, there will be more detailed 4 constraints to be encoded into CNF form and these are described by **Definition 6.4 – 6.7**.

Definition 6.4. (Distinct Constraint) If a cell has a number, the cell contains exactly only one number.

$$\bigwedge_{i=1}^{row} \bigwedge_{j=1}^{col} \bigwedge_{k=0}^{8} \bigwedge_{l=k+1}^{9} (\neg x_{ijk} \vee \neg x_{ijl})$$

Definition 6.5. (Number Zero Constraint) If a cell has a number "0", its neighbor cells including itself cannot be black.

$$\bigwedge_{i=1}^{row} \bigwedge_{j=1}^{col} \bigwedge_{b \in Neighbour\ Cells}^{Number\ of\ Neighbour\ Cells} \bigvee_{a \in Neighbour\ Cells}^{1} (\neg x_{ij0} \vee \neg x_a)$$

Definition 6.6. (Number Nine Constraint) If a cell has a number "9", its neighbor cells including itself must be black.

$$\bigwedge_{i=1}^{row} \bigwedge_{j=1}^{col} \bigwedge_{b \in Neighbour\ Cells}^{Number\ of\ Neighbour\ Cells} \bigvee_{a \in Neighbour\ Cells}^{1} (\neg x_{ij0} \vee x_a)$$

Definition 6.7. (Number k Constraint) If a cell has a number "k", exactly "k" of its neighbor cells including itself must be black. There will be two parts in this constraint:

At most "k" cell is black:

$$\bigwedge_{k=1}^{8} \bigwedge_{i=1}^{row} \bigwedge_{j=1}^{col} \bigwedge_{b \in Neighbour\ Cells}^{nCr, n=Number\ of\ Neighbour\ Cells, r=k-1} \bigvee_{a \in Neighbour\ Cells}^{Number\ of\ Neighbour\ Cells-k+1} (x_{ijk} \supset (x_a))$$

At least "k" cell is black:

$$\bigwedge_{k=1}^{8} \bigwedge_{i=1}^{row} \bigwedge_{j=1}^{col} \bigwedge_{a \in Neighbour\ Cells}^{k} \left(\bigwedge_{b \in Neighbour\ Cells, b \neq a}^{Neighbour\ Cells-a} \left(\left(\left(\bigwedge_{\substack{n=k+1, \\ n \neq k}}^{9} \neg x_{ijn} \right) \wedge x_{ijk} \wedge x_a \right) \supset \neg x_b \right) \right)$$

After encoding the given puzzle using the proposed CNF encoding method, CNF will be produced. This CNF is executed by using the SAT solver to produce the solved Fill-a-Pix puzzle. The results of using the proposed CNF encoding to solve the puzzle will be described in the following section.

7. Expected Results

By solving Fill-a-Pix puzzle as a SAT problem, the implemented formulation is expected to solve any given Fill-a-Pix puzzle of any size. The proposed SAT CNF encoding is also evaluated on standard SAT solvers.

```
SATISFIABLE
.. #2 #3 .. .. .0 .. .. ## ##
.. .. .. ## .3 .. .2 ## ## #6
.. .. #5 ## #5 .3 .. #5 #7 #4
.. #4 ## .5 ## #5 .. #6 .. .3
.. ## .4 .. .5 ## #6 ## ## .3
## ## .. .2 ## #5 .. .. ## ##
#4 .. .1 .. ## .. .1 .1 .. ##
#4 .. .1 .. ## .. .1 .. .4 ##
## ## .. .. #6 ## .. .. ## #4
.. #4 #4 ## ## ## ## #4 ## ..
```

Figure 4. Fill-a-Pix Puzzle with Hidden Picture **of "Fruit"** by Proposed SAT CNF Encoding

The result is shown by two Fill-a-Pix puzzles: the first one with the size of 10 x 10 and the second one with the size of 60 x 100. The given Fill-a-Pix puzzle is encoded using the SAT CNF encoding. Figure 4 shows the results of the solved Fill-a-Pix puzzle (10 x 10) using the proposed SAT CNF Encoding which reveals the hidden picture of "Fruit" as well as Figure 5 shows the results of the solved Fill-a-Pix puzzle (60 x 100) using the proposed SAT CNF Encoding which reveals the hidden picture of "Titanic Ship".

Figure 5. Fill-a-Pix Puzzle with Hidden Picture **of "Titanic Ship"** by Proposed SAT CNF Encoding

Figure 6 shows the results of CPU execution times of each solver for different Fill-a-Pix puzzle sizes and levels. The solver could solve the puzzle within the least amount of time is MiniSAT. This is because MiniSAT uses the Conflict Minimization technique which makes the best use of Backjumping and Resolution to minimize the conflicts. This would reduce the search space and increase the efficiency which would make MiniSAT to solve faster.

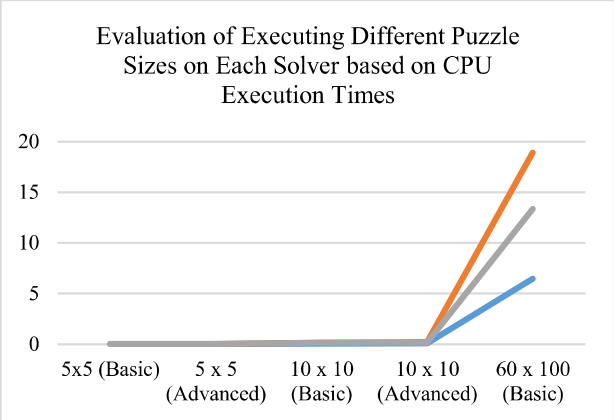

Figure 6. Evaluation of Different Puzzle Sizes and Levels on MiniSAT, CryptoMiniSAT and RSAT

In this research, the number of variables and the number of clauses are strongly dependent on the puzzle sizes. If the puzzles' size is larger, the bigger the number of variables and clauses. Figure 7 shows the difference of the number of variables and clauses as the puzzles' size grows. For the puzzle size greater than 50 x 50, even though the number of variables is 6 thousands, the number of clauses is already around 600 thousands.

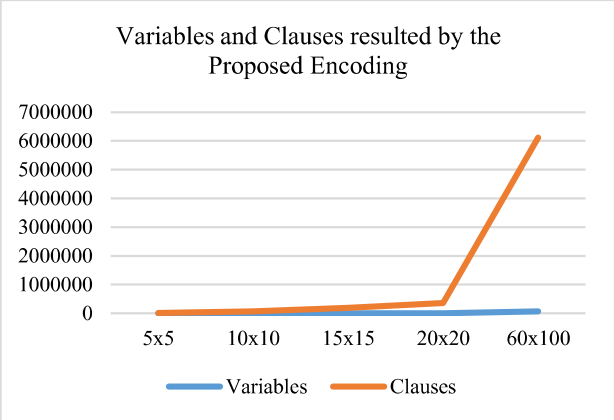

Figure 7. Evaluation of Variables and Clauses on Different Puzzle Sizes

Moreover, the evaluation is presented comparing with the other similar puzzles, too. Sudoku and Slitherlink can be regarded as the most similar puzzle to Fill-a-Pix. Each of them has its own rules and its own way of SAT CNF Encoding. Therefore, the comparison of these puzzles was carried out based on the size of above 50 x 50. The number of variables and clauses produced by the proposed SAT CNF Encoding Method of Fill-a-Pix Puzzle is compared with Sudoku and Slitherlink.

Figure 8 illustrates the comparison of these three puzzles based on the number of variables produced by their own SAT CNF Encoding and can be easily noticed that regardless of how many the puzzle size increases the proposed SAT CNF Encoding Method of Fill-a-Pix Puzzle

could maintain the least variables among the three puzzles. This is concerned with the way of defining variables to the puzzle efficiently.

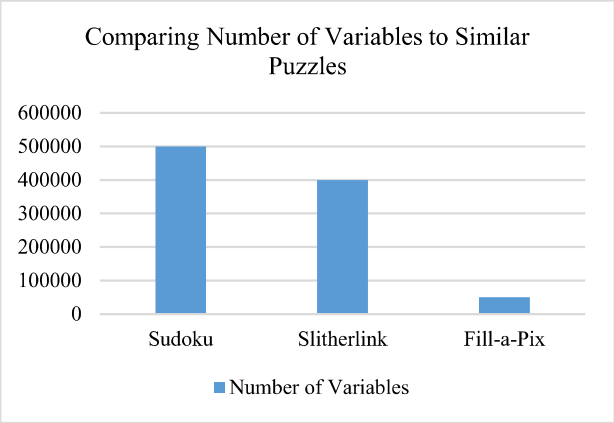

Figure 8. Evaluation the Number of Variables by Sudoku, Slitherlink and Fill-a-Pix

Figure 9 illustrates the comparison of these three puzzles based on the number of clauses produced by their own SAT CNF Encoding. Slitherlink has less clauses than Fill-a-Pix. Even though Fill-a-Pix Puzzle has clauses more than Slitherlink, Fill-a-Pix Puzzle, which has around 5 million clauses, has a greater difference comparing to Sudoku which has nearly 80 million clauses. It is the normal way that as the number of variables bigger, the number of clauses also bigger. However, Slitherlink's clauses number smaller, because it already used the optimization on its SAT CNF Encoding. Fill-a-Pix puzzle also needs to optimize its SAT CNF Encoding as a further research.

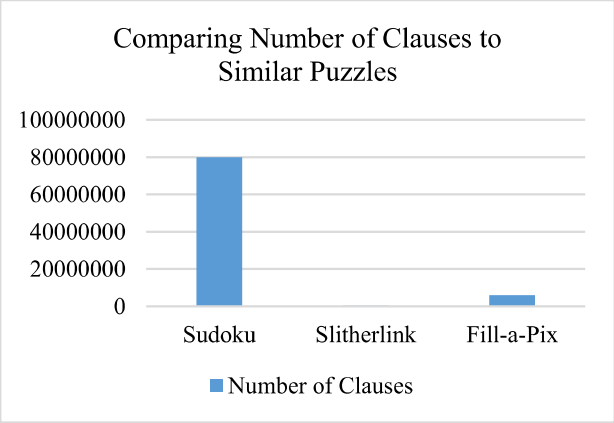

Figure 9. Evaluation the Number of Clauses by Sudoku, Slitherlink and Fill-a-Pix

8. Conclusion and Future Work

In this research paper, Fill-a-Pix puzzle is solved as a SAT problem for the first time and the new SAT CNF encoding is proposed. And, the proposed SAT CNF encoding is evaluated by executing on different SAT solvers – MiniSAT, CryptoMiniSAT and RSAT as well as the results have been compared based on their execution times, variables and clauses.

As a future work, more optimization can be done on the proposed SAT CNF encoding to optimize the execution time of solving the puzzle by applying the heuristic algorithms. In this research paper, only one puzzle i.e. Fill-a-Pix puzzle is evaluated on different kinds of SAT solvers. To advance the studies of SAT problems, different kinds of puzzles could be solved and evaluated on different SAT solvers by comparing their execution times.

9. References

[1] Dennis Yurichev, "SAT/SMT by Example", 25 Jan 2019.

[2] Tjark Weber, "A SAT-based Sudoku Solver", *In 12th International Conference on Logic for Programming, Artificial Intelligence and Reasoning LPAR, pages 11 – 15*, 2005.

[3] Inês Lynce, and Joël Ouaknine, "Sudoku as a SAT Problem", *In International Symposium on Artificial Intelligence and Mathematics ISAIM*, 2006.

[4] Gao Shan, and Roland Yap, "Solving Puzzles (EG. Sudoku) and Other Combinatorial Problems with SAT", *ISAIM*, 2006.

[5] Gihwon Kwon, and Himanshu Jain, "Optimized CNF Encoding for Sudoku Puzzles", 2005.

[6] Shuai Wang, and Aashish Venkatesh, "A SAT Attack on Killer Sudoku Problems", 22 Sep 2016.

[7] Putranto Utomo, and Ruud Pellikaan, "Binary Puzzle as a SAT Problem", 01 Jan 2017.

[8] David Westreicher, "Slitherlink Reloaded", 16 Nov 2011.

[9] Carlos Ansótegui, Ramón Béjar, César Fernández and Carles Mateu, "On the Hardness of Solving Edge Matching Puzzles as SAT or CSP Problems", 22 Sep 2012.

[10] Boolean Satisfiability Problem, Web Page, https://en.wikipedia.org/wiki/Boolean_satisfiability_problem.

[11] Fill-a-Pix Picture Logic Puzzle, Web Page, https://www.conceptispuzzles.com/index.aspx?uri=puzzle/fill-a-pix.

[12] MiniSAT SAT Solver, Web Page, http://minisat.se/.

[13] CryptoMiniSAT SAT Solver, Web Page, https://www.msoos.org/cryptominisat5/.

[14] RSAT SAT Solver, Web Page, http://reasoning.cs.ucla.edu/rsat/home.html.