The Thirty-Second AAAI Conference
on Artificial Intelligence (AAAI-18)

Exploring the Use of Shatter for AllSAT Through Ramsey-Type Problems

David E. Narváez
Golisano College of Computing and Information Sciences
Rochester Institute of Technology
Rochester, NY, USA 14623
den9562@rit.edu

Abstract

In the context of SAT solvers, `Shatter` is a popular tool for symmetry breaking on CNF formulas. Nevertheless, little has been said about its use in the context of ALLSAT problems. ALLSAT has gained much popularity in recent years due to its many applications in domains like model checking, data mining, etc. One example of a particularly transparent application of ALLSAT to other fields of computer science is computational Ramsey theory. In this paper we study the effect of incorporating `Shatter` to the workflow of using Boolean formulas to generate all possible edge colorings of a graph avoiding prescribed monochromatic subgraphs. We identify two drawbacks in the naïve use of `Shatter` to break the symmetries of Boolean formulas encoding Ramsey-type problems for graphs.

1 Introduction

The ALLSAT problem is a variant of the SAT problem where we are interested in finding all the models (satisfying assignments) of the input formula. The survey by Toda and Soh (2016) summarizes the state-of-the-art in techniques used for solving ALLSAT problems. Many ideas from SAT solvers are applicable to ALLSAT solvers with few adaptations. A technique that has proved effective in SAT solvers and is of particular importance for ALLSAT problems is symmetry breaking in Boolean formulas (Sakallah 2009). The survey by Walsh (2012) provides a good overview of the current approaches and techniques used to deal with this problem. `Shatter` (Aloul, Sakallah, and Markov 2006) is a tool that generates *symmetry-breaking clauses* as a preprocessing step to solve Boolean formulas in order to simplify the search space for conventional SAT solvers. `Shatter` has become a popular preprocessing tool since it can be used on any CNF formula encoded in the popular DIMACS format and it has the desirable property that the size of the symmetry-breaking clauses it adds to the formula is linear in the number of variables of the original formula.

To study the effect of `Shatter`'s symmetry breaking approach in a clean application where symmetries can be formally defined and studied, we look at symmetry breaking for Ramsey-type problems in combinatorial computing. In this paper we focus on graph 2-colorings, which are partitions of the edges of a graph into two sets. We say that a graph F *arrows* the pair of graphs (G, H), written $F \rightarrow (G, H)$, when any 2-coloring of the set of edges of a graph F yields a monochromatic G in the first color or a monochromatic H in the second color. It is straightforward to see how 2-color arrowing problems can be encoded into Boolean satisfiability problems. In the case of finite Ramsey numbers, a didactic description of this encoding appears in (Zhang 2009). Given Shatter's popularity, it may seem as a good tool to tackle the generation of irredundant sets of colorings for Ramsey-type problems. Nevertheless, in this paper we identify and discuss some drawbacks of using this approach without caution. Our technical report (Narváez 2017) provides strategies and additional tools to cope with the negative effects of using `Shatter` for ALLSAT.

2 Background and Definitions

A fundamental construction in `Shatter`'s formulation of symmetry breaking predicates is a graph that encodes the relationship between clauses and literals of the input formula. The symmetry breaking predicates added by `Shatter` come from the automorphisms of this graph. A bird's eye view of the process `Shatter` uses to break the symmetries of a Boolean formula ϕ is as follows: (a) the graph G_ϕ is generated as per the rules in (Aloul, Sakallah, and Markov 2006), (b) the group of automorphisms of G_ϕ is found, (c) a subset of these automorphisms (in particular, `Shatter` uses the generators of the group, following (Crawford et al. 1996)) is used to generate symmetry breaking clauses that are added to ϕ .

2.1 Ramsey Colorings

Several Ramsey-type problems can be expressed in terms of the arrowing property defined in Section 1. In particular, the generalized Ramsey number $R(G, H)$ can be defined as the smallest natural number N such that K_N arrows the pair (G, H). We define $\mathcal{C}(F; G, H)$ as the set of colorings witnessing $F \not\rightarrow (G, H)$. Clearly, $\mathcal{C}(F; G, H) = \emptyset \Leftrightarrow F \rightarrow (G, H)$. We will call a set of colorings for parameters F, G and H *complete* if every isomorphism class in $\mathcal{C}(F; G, H)$ is represented in the set. To use ALLSAT solvers to generate graph 2-colorings, we exploit the fact that the Boolean domain contains two values $\perp$ (*false*) and $\top$ (*true*) and express the negation of the arrowing property

$$\phi(F;G,H) = \left(\bigwedge_{s \in \mathcal{S}(F,G)} \bigvee_{\{u,v\} \in E(G)} x_{\{s(u),s(v)\}} \right) \wedge \left(\bigwedge_{s \in \mathcal{S}(F,H)} \bigvee_{\{u,v\} \in E(H)} \overline{x_{\{s(u),s(v)\}}} \right) \quad (1)$$

Figure 1: The standard encoding for the non-arrowing property as a Boolean formula. Here, $\mathcal{S}(X,Y)$ denotes the set of non-induced subgraph isomorphisms from X to Y (see Section 2).

in terms of Boolean formulas. Consider the Boolean formula $\phi(F;G,H)$ on variables $x_{\{u,v\}}$ for every edge $\{u,v\}$ in F defined as per equation (1) in Figure 1. One can generate $\mathcal{C}(F;G,H)$ by using an ALLSAT solver to list every model of $\phi(F;G,H)$. An undesirable characteristic of this approach is that $\mathcal{C}(F;G,H)$ can be large and one is in general more interested in generating unique (under isomorphism) witness colorings directly from the SAT formulation of $F \not\to (G,H)$.

3 Results

One of the main improvements of `Shatter` over the original formulation of the symmetry breaking clauses (Crawford et al. 1996) is that `Shatter` adds symmetry-breaking clauses whose number of literals is linear in the number of variables of the input formula. This is achieved by chaining the symmetry breaking constraints and adding a relaxation. This relaxation has an undesirable effect in the number of satisfying assignments of the resulting formula. In (Narváez 2017) we present a detailed analysis of this effect.

To illustrate this issue we provide a concrete example. From finite Ramsey theory, we know that $R(C_5,C_5) = 9$ (Chartrand and Schuster 1971), where C_5 is the cycle of length 5. This means that $\phi(K_9;C_5,C_5) \notin$ SAT, but $\phi(K_8;C_5,C_5) \in$ SAT, so we are interested in finding all edge colorings of the complete graph K_8 witnessing $R(C_5,C_5) > 8$. $\phi(K_8;C_5,C_5)$ contains 28 variables (corresponding to $\binom{8}{2}$ edges in K_8) and 1344 clauses, and there are 1190 models for that formula. From this information, we know that $|\mathcal{C}(K_8;C_5,C_5)| = 1190$. After processing this formula with Shatter, the resulting formula with symmetry breaking clauses has 70 variables, 1499 clauses, and 824 models. On the other hand, using our own implementation of the chaining method without the relaxation outputs a formula with 165 variables, 1809 clauses, and 5 models. Using `nauty` (McKay and Piperno 2014) to reduce any of these sets of colorings to pick just one representative from each equivalence class of colorings under isomorphism, we find that there are 4 unique colorings, so the chaining method without the relaxation outputs only one redundant coloring.

A sufficient condition for the set of colorings generated from the formula output by Shatter to be incomplete is the presence of *free variables*. For details, see (Narváez 2017), where we also provide an example of a graph G_{ex} for which the coloring set generated from the output of preprocessing $\phi(G_{ex},C_5,C_5)$ with Shatter does not generate a complete set of colorings even though $\phi(G_{ex},C_5,C_5)$ has no free variables. This shows that the condition we present is not a necessary one.

4 Conclusion

While `Shatter` has been an influential tool in the field of symmetry breaking in Boolean formulas for over a decade, this paper shows that it has problems when applied to ALLSAT. Recently, `BreakID` (Devriendt et al. 2016) has built upon the symmetry breaking techniques introduced by `Shatter` and has added some novel ideas like row interchangeability. Even though `BreakID` implements the same relaxations as `Shatter`, it does include an option to not use these relaxations and is thus better suited for ALLSAT applications since it will not introduce additional models.

Acknowledgments

The author would like to thank the anonymous reviewers of the Thirty-Second AAAI Conference on Artificial Intelligence (AAAI-18) for their valuable comments.

References

Aloul, F. A.; Sakallah, K. A.; and Markov, I. L. 2006. Efficient symmetry breaking for boolean satisfiability. *IEEE Transactions on Computers* 55(5):549–558.

Chartrand, G., and Schuster, S. 1971. On the existence of specified cycles in complementary graphs. *Bulletin of the American Mathematical Society* 77:995–998.

Crawford, J. M.; Ginsberg, M. L.; Luks, E. M.; and Roy, A. 1996. Symmetry-breaking predicates for search problems. In *International Conference on the Principles of Knowledge Representation and Reasoning*, 148–159. Morgan Kaufmann.

Devriendt, J.; Bogaerts, B.; Bruynooghe, M.; and Denecker, M. 2016. Improved static symmetry breaking for SAT. In *SAT*, volume 9710 of *Lecture Notes in Computer Science*, 104–122. Springer.

McKay, B. D., and Piperno, A. 2014. Practical graph isomorphism, II. *Journal of Symbolic Computation* 60:94–112.

Narváez, D. 2017. Exploring the use of Shatter for ALLSAT through Ramsey-type problems. Technical Report arXiv:1711.06362 [cs.AI], Computing Research Repository.

Sakallah, K. A. 2009. Symmetry and satisfiability. In *Handbook of Satisfiability*, volume 185 of *Frontiers in Artificial Intelligence and Applications*. IOS Press. 289–338.

Toda, T., and Soh, T. 2016. Implementing efficient all solutions SAT solvers. *Journal of Experimental Algorithmics* 21:1.12:1–1.12:44.

Walsh, T. 2012. Symmetry breaking constraints: Recent results. In *Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence*, 2192–2198. AAAI Press.

Zhang, H. 2009. Combinatorial designs by SAT solvers. In *Handbook of Satisfiability*, volume 85 of *Frontiers in Artificial Intelligence and Applications*. IOS Press. 533–568.