Bitwuzla-MachBV at SMT-COMP 2025

Xiang Zhang[1,2], Shaohuang Chen[1,2], Mengyu Zhao[1,2], and Shaowei Cai[1,2]

[1] Key Laboratory of System Software (Chinese Academy of Sciences) and State Key Laboratory of Computer Science, Institute of Software, Chinese Academy of Sciences, Beijing, China
{zhangx, chensh, zhaomy, caisw}@ios.ac.cn
[2] University of Chinese Academy of Sciences, Beijing, China

1 Introduction

Bitwuzla-MachBV is a **derived** tool based on Bitwuzla[2](v0.7.0). It participates in the Single Query Track of the QF_BV logic. You can find the tool and experimental data that we have prepared at Bitwuzla-MachBV-at-SMT-COMP-2025.

Bitwuzla-MachBV comprises two primary components: the strategy generator MachBV and the base solver Bitwuzla.

1. MachBV is a lightweight SMT-BV solver strategy generation system, which is implemented in C++. MachBV adopts a modular design and uses binary classifiers based on XGBoost[1] to generate customized solving strategies for SMT(QF_BV) instances.
2. Bitwuzla is a high performance SMT solver for the theories of fixed-size bit-vectors, floating-point arithmetic, arrays and uninterpreted functions and their combinations. For SMT-COMP 2025, we use **version 0.7.0** as the base solver. You can find it on GitHub at Bitwuzla v0.7.0
3. XGBoost is an optimized distributed gradient boosting library designed to be highly efficient, flexible and portable. It implements machine learning algorithms under the Gradient Boosting framework. XGBoost provides a parallel tree boosting (also known as GBDT, GBM) that solve many data science problems in a fast and accurate way. For SMT-COMP 2025, we use **version 3.0.2** as the base classifier. You can find it on GitHub at XGBoost v3.0.2.

2 Framework

MachBV is composed of three main modules: syntax parsing, feature extraction, and expert decision-making.

Syntax Parsing: transforms SMT-BV problems into directed acyclic graphs (DAGs), preserving the semantic structure and removing redundant information.

Feature Extraction: analyzes these DAGs to generate feature vectors that reflect the syntactic structure of the problems. This module extracts structural features of nodes with category information in DAGs by calculating in-degree

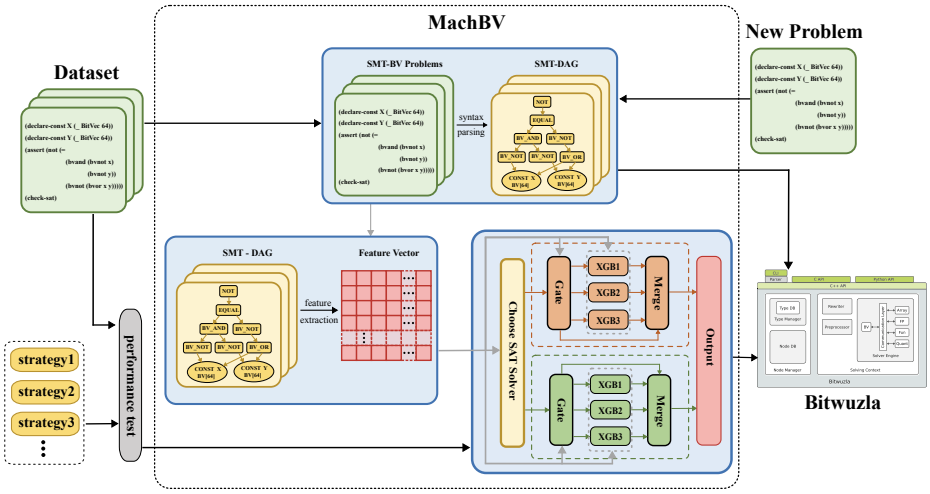

Fig. 1. MachBV framework.

statistics by category. Specifically, the k-th dimension of the feature vector represents the sum of in-degrees of all nodes categorized as k in the DAG.

Expert Decision-making: uses these feature vectors, applying domain classification, gating, and expert submodules to determine the optimal solving strategy. We discovered complementarity among different preprocessing algorithms and even between different SAT solvers. By fully leveraging these complementary advantages, we can significantly enhance the solver's performance. Therefore, we designed this module to attempt to assign an appropriate solving strategy for each problem.

3 Dataset

Training data is well known to have a major impact on the performance of machine learning methods. For QF_BV theory, the SMT-LIB benchmark serves as the most important data source, and we inevitably use train data from this collection. This raises a potential concern when using MachBV in SMT-COMP: since the training data also come from the SMT-LIB benchmark, how can we avoid suspicions that MachBV directly memorizes answers based on syntactic structures?

To address this concern, we carefully filtered our training data. Specifically, we first run Bitwuzla v0.7.0 on the entire benchmarks and recorded its performance on each problem. We retained only those problems that Bitwuzla successfully solved within 1200 seconds and discarded the rest, then used the remaining problems to train MachBV. The detailed performance results of Bitwuzla on these benchmarks, as well as the final list of selected training problems, are available at Train Data.

We adopted this approach because SMT-COMP rankings primarily consider the number of successfully solved problems. By removing the critical problems that could determine rankings, we ensure that MachBV's training data do not include problems that the original Bitwuzla could not solve. This demonstrates that MachBV's improvement over Bitwuzla's solving effectiveness is not because MachBV was trained on problems that Bitwuzla could not solve, but rather because MachBV learned relevant knowledge from the remaining problems. Additionally, the test data used in SMT-COMP are scrambled, and this scrambling can alter the extracted features; different random seeds for scrambling may lead to different feature extraction results.

4 Acknowledgements

We acknowledge the authors of Bitwuzla for their foundational work, which our derived tool is built upon. The Bitwuzla solver they developed provides a rich set of solving strategy configuration options, enabling us to test the performance of various strategy configurations and discover complementarities between different strategy settings. Additionally, Bitwuzla's inherent high performance ensures that Bitwuzla-MachBV achieves excellent results.

Finally, we would like to express our gratitude once again to the authors of Bitwuzla.

References

1. Chen, T., Guestrin, C.: Xgboost: A scalable tree boosting system. In: Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining. pp. 785–794 (2016)
2. Niemetz, A., Preiner, M.: Bitwuzla. In: Enea, C., Lal, A. (eds.) Computer Aided Verification - 35th International Conference, CAV 2023, Paris, France, July 17-22, 2023, Proceedings, Part II. Lecture Notes in Computer Science, vol. 13965, pp. 3–17. Springer (2023). https://doi.org/10.1007/978-3-031-37703-7_1, `https://doi.org/10.1007/978-3-031-37703-7_1`