Murxla: A Modular and Highly Extensible API Fuzzer for SMT Solvers

Aina Niemetz, Mathias Preiner and Clark Barrett

CAV, August 8, 2022

Satisfiability Modulo Theories (SMT) Solvers

- **Tools to solve the SMT Problem**
  - complex and large pieces of software
    - Bitwuzla: $\sim$ 90k LOC
    - cvc5: $\sim$ 300k LOC
    - z3: $\sim$ 500k LOC
  - back-ends in higher-level tool chains
  - **strong requirements:**
    - performance
    - robustness
    - correctness
  - **traditional testing:**
    - unit testing
    - maintaining a regression test suite
    - insufficient for achieving high levels of robustness
  - random stress testing (**fuzzing**)

Fuzz Testing SMT Solvers

SMT solvers provide **two** interfaces:

full knowledge of input structure

- textual interface (SMT-LIB)
  - input fuzzing
  - ▷ generate valid SMT-LIB input
  - \+ significantly less effort
  - − no solver-specific features
- application programming interface (API)
  - API fuzzing
  - ▷ generate valid sequences of solver API calls
  - ▷ link against solver library
  - \+ solver-specific features
  - \+ subsumes input fuzzing (except parser)
  - − more involved

Murxla

... a model-based API Fuzzer for SMT solvers

- lifts grammar-based input fuzzing to API level

- Semantic (data) model
  - defines constructs (theories, sorts, operators, commands)
  - based on SMT-LIBv2
- API model
  - defines the usage of the solver API itself
- Options model
  - defines solver configuration options and valid combinations

What do we consider a bug?

- soundness issues
  - solver answers *unsat* when input is *sat*
  - solver answers *sat* when input is *unsat*
- crashes (assertion failures, segmentation faults, ...)

Murxla

... a model-based API Fuzzer for SMT solvers

- Model-based **API fuzzer**
  - ▷ generates valid sequences of solver API calls
  - ▷ general enough to support any SMT solver
  - ▷ highly extensible to support all solver-specific features
- **Tracer**
  - ▷ records API call sequences as an API trace
- **Untracer**
  - ▷ replays API traces to reproduce original behavior
- **Delta Debugger**
  - ▷ minimizes API traces while preserving the original behavior

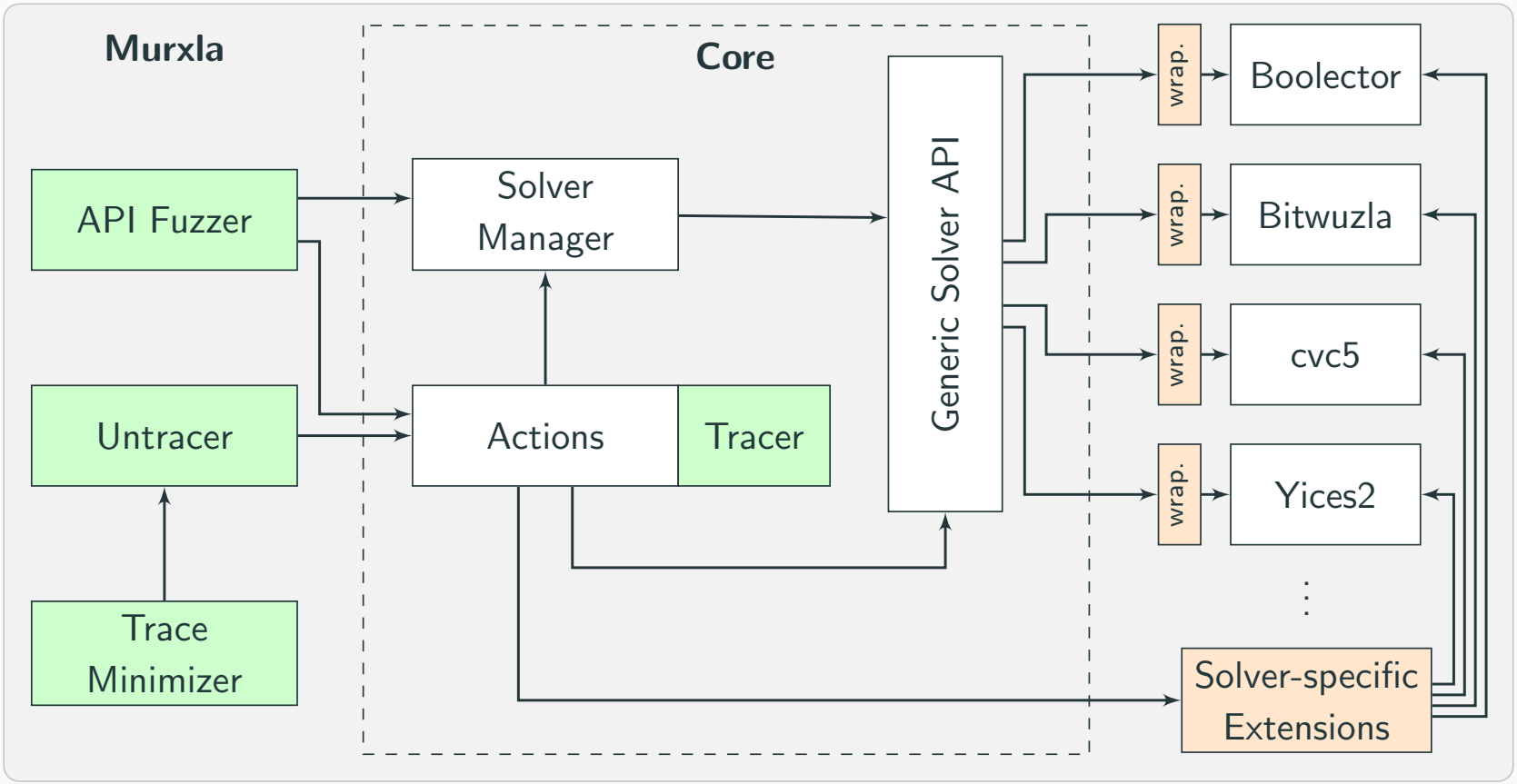
* Provided they allow being integrated into a C++ tool.

Murxla

... a model-based API Fuzzer for SMT solvers

- **translate** API traces to **SMT-LIBv2**
  - if trace doesn't contain solver-specific extensions
  - especially useful for minimized traces
  - can then be further reduced with ddSMT [4]
- **generate** SMT-LIBv2 input
  - can be used as SMT-LIB input fuzzer with any solver binary
- **cross-check** two solver instances
  - two integrated solvers under test
  - one integrated solvers vs. a solver via the SMT-LIBv2 interface

Murxla Architecture



Evaluation

Murxla vs. BtorMBT (Boolector)

Murxla			BtorMBT [1]		
L [%]	F [%]	I [#]	L [%]	F [%]	I [#]
81.1	87.5	18	72.3	80.6	0

Murxla vs. Input Fuzzers (cvc5, QF_SLIA)

Murxla			Storm [2]			Murxla-cc			TypeFuzz [3]		
L [%]	F [%]	I [#]	L [%]	F [%]	I [#]	L [%]	F [%]	I [#]	L [%]	F [%]	I [#]
37.8	52.5	7	20.2	34.3	0	21.5	36.3	1	17.4	30.8	0

I ... Number of issues
F ... Function coverage
L ... Line coverag

Murxla-cc ... cross-checking configuration (Z3 vs cvc5)

1 hour, with 1 second time limit per round

Conclusion

- open source
  - https://github.com/murxla/murxla
  - implemented in C++
  - GPL-v3.0 license

- comprehensive documentation available at
  - https://murxla.github.io

References

- A. Niemetz, M. Preiner and A. Biere. Model-Based API Testing for SMT Solvers. In Proc. of SMT'17, pages 3–14, 2017. http://ceur-ws.org/Vol-1889/paper1.pdf
- M. N. Mansur, M. Christakis, V. Wüstholz and F. Zhang. Detecting critical bugs in SMT solvers using blackbox mutational fuzzing. In Proc. of ESEC/FSE'20, pages 701–712, ACM, 2020. https://doi.org/10.1145/3368089.3409763
- J. Park, D. Winterer, C. Zhang and Z. Su. Generative type-aware mutation for testing SMT solvers. In Proc. of OOPSLA'21, pages 1–19, ACM, 2021. https://doi.org/10.1145/3485529
- G. Kremer, A. Niemetz and M. Preiner. ddSMT 2.0: Better Delta Debugging for the SMT-LIBv2 Language and Friends. In Proc. of CAV'21, pages 231–242, Springer, 2021. http://dx.doi.org/10.1007/978-3-030-81688-9_11