Smt-Switch:
A Solver-Agnostic C++ API for SMT Solving

Makai Mann, Amalee Wilson, Yoni Zohar, Lindsey Stuntz,
Ahmed Irfan, Kristopher Brown, Caleb Donovick,
Allison Guman, Cesare Tinelli, Clark Barrett

SMT Workshop 2021
Presentation-Only
Appearing at SAT 2021

Motivation

- Many high-quality SMT solvers with different strengths
  - Often implemented in C/C++

- Interactive use common
  - Queries depend on previous results (dynamic querying)
  - Manipulate, traverse, and rewrite terms

Motivation

- Typical Approaches for C++ Tools
  - Pick a specific solver and API
  - Communicate via pipes with SMT-LIB

- Goal: Provide generic, high-quality access to various SMT solvers

Smt-Switch Design

- Abstract interfaces
  - Implemented via inheritance by different solver backends

- Nomenclature
  - *Underlying solver*: a specific SMT solver (e.g., CVC4)
  - *Backend:* Smt-Switch implementation for an underlying solver

Architecture Overview

- Abstract classes
  - AbsSort
  - AbsTerm
  - AbsSmtSolver

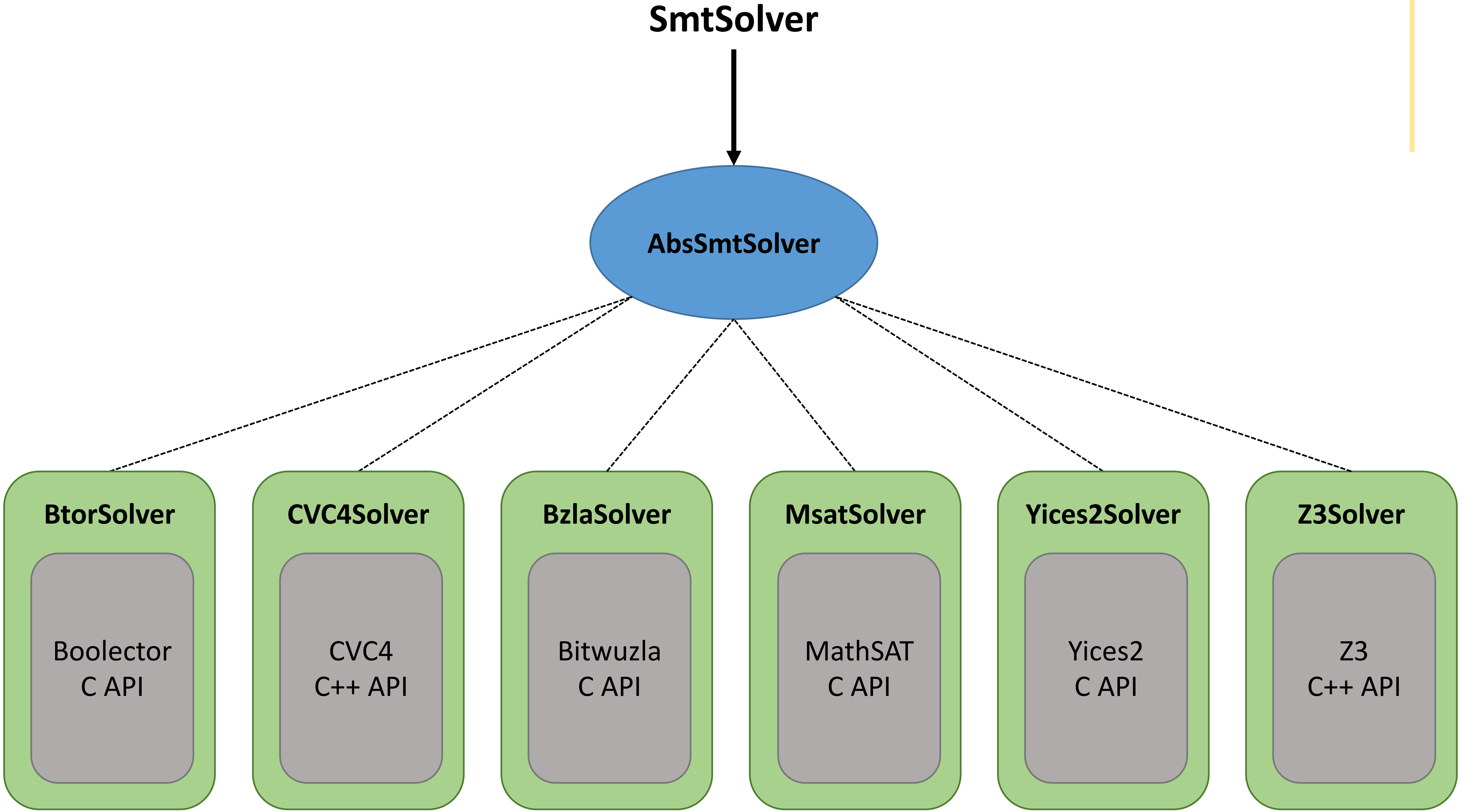
- Structs
  - Op
  - Result

Architecture Overview

- Abstract classes
  - AbsSort
  - AbsTerm
  - AbsSmtSolver

- Smart Pointers
  - ← Sort
  - ← Term
  - ← SmtSolver

- Structs
  - Op
  - Result



SmtSolver

AbsSmtSolver

BtorSolver

Boolector
C API

CVC4Solver

CVC4
C++ API

BzlaSolver

Bitwuzla
C API

MsatSolver

MathSAT
C API

Yices2Solver

Yices2
C API

Z3Solver

Z3
C++ API

Example

Notable Additional Utilities

- TermTranslator: transfers terms between solver instances
- GenericSolver: communicates via pipes with arbitrary binary
- PrintingSolver: dumps SMT-LIB for API commands
- Portfolio Solving: runs multiple solver instances in parallel
- SmtLibReader: parser for subset of SMT-LIB
  - No support for datatypes, floating point, or strings
- Cython-based Python bindings
- PySMT frontend: transfers terms to and from PySMT

Experiments

- Ran each solver binary alone vs corresponding Smt-Switch backend

- Ran on BV and ABV benchmarks since all solvers support
  - All existing combinations of incremental/non-incremental and quantified/QF

- 66,691 total
  - Suggests < 10% overhead on practical problems
  - Rough approximation because different parser implementations

Conclusion

- C++ API for SMT Solving
  - Based on SMT-LIB
  - Simple design for easy maintenance and extension
- Currently supports
  - Boolector
  - Bitwuzla
  - CVC4
  - MathSAT
  - Yices2
  - Z3
- On GitHub: https://github.com/makaimann/smt-switch