

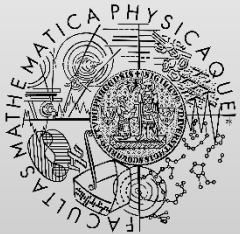
Automated Reasoning

<http://d3s.mff.cuni.cz>

Department of
Distributed and
Dependable
Systems



Pavel Parízek



FACULTY
OF MATHEMATICS
AND PHYSICS
Charles University

Main categories of tools

- SAT solvers
- SMT solvers
- Theorem provers

SAT solvers

- Domain: propositional logic
 - Formulas over boolean variables
- Tools
 - MiniSAT (<http://minisat.se/>)
 - Lingeling (<http://fmv.jku.at/lingeling/>)
 - Glucose (<https://www.labri.fr/perso/lsimon/glucose/>)
- Applications
 - Hardware & software verification (testing)
 - Efficiently solving various problems encoded to SAT

SMT solvers

- Domain: first-order predicate logic with specific theories and other restrictions
 - Formulas include predicates and functions
 - arithmetic expressions (+, -)
 - relational operators (=, >, <)
 - Theories: linear arithmetic, bitvectors, arrays, strings
 - Restrictions: limited support for quantifiers
- Tools
 - Z3 (<https://github.com/Z3Prover/z3>)
 - CVC5 (<https://cvc5.github.io/>)
 - OpenSMT (<http://verify.inf.usi.ch/opensmt>)
 - Common input format: SMT-LIB

Theorem proving

- Domain: complete first-order predicate logic
 - Mathematical induction
 - Higher-order logic (HOL)
 - Machine-checked proofs
- Very powerful, but only partially automated
 - Interactive (requires human assistance)
- Input: set of axioms (theory T), general formula ϕ
 - Relevant use case: proof obligations
- Tools: PVS, Isabelle/HOL, Rocq

Theorem proving – tools

- PVS
 - <https://pvs.csl.sri.com/>
- Isabelle/HOL
 - <https://isabelle.in.tum.de/>
- Rocq
 - <https://rocq-prover.org/>

PVS – introduction

- Download from <https://pvs.csl.sri.com/downloads.html> and install
 - Version: PVS 7.1, Linux allegro 64-bit
 - How: unpack & run `install-sh`
- Running: `./pvs`
- Important commands
 - Quit the PVS environment: `Ctrl-x Ctrl-c`
 - Help: `Ctrl-c h` //leave by typing “q”
- Basic guide
 - <https://pvs.csl.sri.com/doc/pvs-system-guide.pdf>

PVS – usage (commands)

- Opening file: Ctrl-x Ctrl-f
- Switch buffer (file): Ctrl-x b
- Close buffer: type character “q”
- Demo 1: sum.pvs
 - Type checking (show that function is total)
 - Commands: Alt-x tc, Alt-x tcp
 - Proving main theorem semi-automatically
 - Approach: traverse all branches in the proof tree
 - Start by PVS command: Alt-x pr
 - Relevant prover commands: (induct “n”), (expand “sum”), (assert), (skolem!), (flatten)
- Demo 2: stacks.pvs
- Demo 3: fm99/phone_1.pvs

Theorem proving – other tools

- The KeY Project

- <https://www.key-project.org/>

- ACL2

- <https://www.cs.utexas.edu/users/moore/acl2/>

- Lean

- <https://leanprover.github.io/>

Related courses

- Decision Procedures and Verification (NAIL094)
 - <http://ktiml.mff.cuni.cz/~kucera/satsmt/index-en.php>
- Formal Mathematics and Proof Assistants
 - NMMB568, LS 2025/26, Department of Algebra

Runtime verification (bonus topic)

- Monitors
 - Recording interesting events
 - field accesses, method calls, thread synchronization
 - Checking functional correctness properties defined as finite state machines
- Further details
 - https://en.wikipedia.org/wiki/Runtime_verification
- State of the art: conference RV
 - <https://runtime-verification.github.io/events/>