Engineering Theories with Z3

Nikolaj Bjørner

Microsoft Research
nbjorner@microsoft.com

Abstract

Modern Satisfiability Modulo Theories (SMT) solvers are fundamental to many program analysis, verification, design and testing tools. They are a good fit for the domain of software and hardware engineering because they support many domains that are commonly used by the tools. The meaning of domains are captured by theories that can be axiomatized or supported by efficient *theory solvers*. Nevertheless, not all domains are handled by all solvers and many domains and theories will never be native to any solver. We here explore different theories that extend Microsoft Research's SMT solver Z3's basic support. Some can be directly encoded or axiomatized, others make use of user theory plug-ins. Plug-ins are a powerful way for tools to supply their custom domains.

1 Introduction

This talk surveys a selection of theories that have appeared in applications of Z3 [5] and also in recent literature on automated deduction. In each case we show how the theories can be supported using either existing built-in theories in Z3, or by adding a custom decision procedure, or calling Z3 as a black box and adding axioms between each call. The theme is not new. On the contrary, it is very central to research on either encoding (reducing) theories into a simpler basis or developing special solvers for theories. Propositional logic is the most basic such basis e.g., [6]. In the context of SMT (Satisfiability Modulo Theories), the basis is much richer. It comes with built-in support for the theory of equality, uninterpreted functions, arithmetic, arrays, bit-vectors, and even first-order quantification. The problem space is rich, and new applications that require new solutions keep appearing. We don't offer a silver bullet solution, but the "exercise" of examining different applications may give ideas how to tackle new domains.

Z3 contains an interface for plugging in custom theory solvers. We exemplify this interface on two theories: MaxSMT and partial orders. This interface is powerful, but also requires thoughtful interfacing. To date it has been used in a few projects that we are aware of [8, 1, 7]. Some of our own work can also be seen as an instance of a theory solver. The quantifier-elimination procedures for linear arithmetic and algebraic data-types available in Z3 acts as a special decision procedure [2]. The OpenSMT solver also supports an interface for pluggable theories [4]. We feel that the potential for plugging in custom theory solvers into modern SMT solvers is enormous.

Z3 also allows interfacing theories in simpler ways. The simplest is by encoding a theory using simpler theories and often also first-order quantification. We discuss two encodings for a theory of object graphs. A usage model that lies between encoding and a user theory, is by calling Z3 repeatedly. Whenever Z3 returns a satisfiable state, then add new axioms that are not satisfied by the current candidate model for the existing formulas. A theory of Higher-Order Logic, HOL, can be encoded using this approach.

Code samples illustrating the theory integrations are available in F# from `http://research.microsoft.com/en-us/events/z3dtu/usertheories.zip`. An extended version of this abstract appears in [3].

References

[1] Anindya Banerjee and David Naumann and Stan Rosenberg. Decision Procedures for Region Logic. *In submission*, Aug. 2011. http://www.cs.stevens.edu/ naumann/publications/dprlSubm.pdf.

[2] N. Bjørner. Linear Quantifier Elimination as an Abstract Decision Procedure. In J. Giesl and R. Hähnle, editors, *IJCAR*, volume 6173 of *Lecture Notes in Computer Science*, pages 316–330. Springer, 2010.

[3] N. Bjørner. Engineering Theories with Z3. In H. Yang, editor, *APLAS*, volume 7078 of *Lecture Notes in Computer Science*, pages 4–16. Springer, 2011.

[4] R. Bruttomesso, E. Pek, N. Sharygina, and A. Tsitovich. The opensmt solver. In J. Esparza and R. Majumdar, editors, *TACAS*, volume 6015 of *Lecture Notes in Computer Science*, pages 150–153. Springer, 2010.

[5] L. M. de Moura and N. Bjørner. Z3: An efficient smt solver. In C. R. Ramakrishnan and J. Rehof, editors, *TACAS*, volume 4963 of *Lecture Notes in Computer Science*, pages 337–340. Springer, 2008.

[6] S. K. Lahiri, S. A. Seshia, and R. E. Bryant. Modeling and verification of out-of-order microprocessors in uclid. In M. Aagaard and J. W. O'Leary, editors, *FMCAD*, volume 2517 of *Lecture Notes in Computer Science*, pages 142–159. Springer, 2002.

[7] P. Rümmer and C. Wintersteiger. Floating-point support for the Z3 SMT Solver. `http://www.cprover.org/SMT-LIB-Float`.

[8] P. Suter, R. Steiger, and V. Kuncak. Sets with cardinality constraints in satisfiability modulo theories. In R. Jhala and D. A. Schmidt, editors, *VMCAI*, volume 6538 of *Lecture Notes in Computer Science*, pages 403–418. Springer, 2011.