Decision Procedures for Containers

Deliverable D1-2

ANR project VECOLIB

November 2015

Abstract

This deliverable presents the decision procedures developed inside the VECOLIB project for reasoning about implementation and content of containers.

Verification of programs implementing containers requires reasoning about complex, unbounded size data structures that may carry data ranging over infinite domains. Examples of such structures are multi-linked lists, nested lists, trees, etc, see deliverable 4.1 of Vecolib. These implementations perform operations that may modify the shape (due to dynamic creation and destructive updates) as well as the data attached to the elements of such data structures. An important issue is the design of logic-based frameworks that express assertions about program configurations, and then allows to check automatically the validity of these assertions, for all computations. This leads to the challenging problem of finding relevant compromises between expressiveness, automation, and scalability. The frameworks and tools developed in the Vecolib project provide solutions to this problem.

Decision procedures for Separation Logic with integer constraints

The first solution, published in [2], is based on Separation Logic (SL). Separation Logic with inductive definitions (SLID) is a well-known approach for deductive verification of programs that manipulate dynamic data structures. Deciding verification conditions in this context is usually based on user-provided lemmas relating the inductive definitions (ID). We propose a novel approach for generating these lemmas automatically which is based on simple syntactic criteria and deterministic strategies for applying them. Our approach focuses on iterative programs, although it can be applied to recursive programs as well, and specifications that describe not only the shape of the data structures, but also their content or their size.

This approach is based on a new class of inductive definitions for describing fragments of data structures that (i) supports very simple lemmas, most of them being extensions of the case where data constraints are not present in the inductive definition, and (ii) allows to automatically synthesise these lemmas using efficiently checkable, almost syntactic, criteria.

In addition to this form of ID, we propose a proof strategy using such lemmas, based on simple syntactic matchings of SL atoms and reductions to SMT solvers for dealing with the data constraints.

We implemented this approach in the SPEN [3] solver and tested it on two sets of benchmarks:

- **RDBI:** 110 verification conditions for proving the correctness of iterative procedures (delete, insert, search) over recursive data structures storing integer data: sorted lists, binary search trees (BST), AVL trees, and red black trees (RBT).
- **SL-COMP'14:** 30 problems in the SL-COMP'14 benchmark, without data constraints, where the inductive definitions are in the class we proposed.

Each problem is solved in less than 1 second, including the call to the SMT solver on the data constraints (described in the next part).

Decision procedure for bag and set constraints

The second decision procedure developed concerns the theory of quantifier free constraints on bags (multi-sets) and sets over integers (QFBILIA). This theory is used not only in the context of the above work on Separation Logic, but it is also the basis of some verifications tools (e.g., Why3, B method). A decision procedure for this theory has been published in [5]. However, there is no SMTLIB theory for this logic and the available SMT solvers (e.g., Z3 [1]) does not deal with such constraints. We submitted this subject to an internship student, Etienne Toussaint, during the summer 2015. He implemented a solver, called BATS, for this theory, with two algorithms. The first algorithm reduces the QFBILIA satisfiability problem to a satisfiability problem in the theory of QFLIA (quantifier free linear integer arithmetics), which is dealt by the current SMT solvers. The second algorithm does a reduction to the QFUFLIA theory, i.e., QFLIA with non-interpreted function symbols, which is also efficiently dealt by SMT solvers. Moreover, these two procedures are optimised to generate small formulas ans so improve the efficiency of SMT solvers.

This solver if formally described in Etienne Toussaint's internship report [4] and the implementation is also available on [3]. BATS has been used in the Spen solver to decide the validity of verification conditions in Separation Logic with data constraints.

References

[1] L. De Moura and N. Bjørner. Z3: An efficient SMT solver. In *Tools and Algorithms for the Construction and Analysis of Systems*, LNCS, pages 337–340. Springer, 2008.

[2] C. Enea, M. Sighireanu, and Z. Wu. On automated lemma generation for separation logic with inductive definitions. In *ATVA*, volume 9364 of *LNCS*, pages 80–96. Springer, 2015. Available as Appendix A.

[3] SPEN. https://www.github.com/mihasighi/spen.

[4] E. Toussaint. Deciding set and multi-set constraints. Internship report, University Paris Diderot, September 2015. Available as Appendix B.

[5] C. G. Zarba. Combining multisets with integers. In *CADE*, LNCS, pages 363–376. Springer, 2002.