Invariant Stream Generators using Automatic Abstract Transformers based on a Decidable Logic[⋆]

Pierre-Loïc Garoche[1,2], Temesghen Kahsai[2] and Cesare Tinelli[2]

[1] Onera, the French Aerospace Lab, France
[2] The University of Iowa

The use of formal analysis tools on system models or code often requires the availability of auxiliary invariants about the studied system. Abstract interpretation is currently one of the best approaches to discover useful invariants, especially numerical ones. However, its application is limited by two orthogonal issues: (i) developing an abstract interpretation is often non-trivial; each transfer function of the system has to be represented at the abstract level, depending on the abstract domain used; (ii) with precise but costly abstract domains, the information computed by the abstract interpreter can be used only once the a post fix point has been reached; something that may take a long time for very large system analysis or with delayed widening to improve precision.

In this work we try to address these issues by combining techniques from abstract interpretation and logic-based model checking. Specifically, we propose a general method for the automatic definition of abstract interpreters that compute numerical invariants of transition systems. We rely on the possibility of encoding the transition system in a decidable logic—such as those typically used by SMT-based model checkers—to compute transformers for an abstract interpreter *completely automatically*. Our method has the significant added benefit that the abstract interpreter can be instrumented to generate system invariants on the fly, during its iterative computation of a post fix point. A prototype implementation of the method provides initial evidence of the feasibility of our approach and the usefulness of its incremental invariant generation feature.

While motivated by practical issues (namely, the generation of auxiliary invariants for a k-induction model checker) the current work is more general and can be adapted to a wide variety of contexts. It only requires that the transition system semantics be expressible in a decidable logics with an efficient solver, such as SAT or SMT solvers, and that the elements of the chosen abstract domain be effectively representable in that logic. Such requirements are satisfied by a large number of abstract domains used in current practice. As a consequence, we believe that our approach could help considerably in expanding the reach of abstract interpretation techniques to a variety of target languages, as well as facilitate their integration with complementary techniques such as model checking ones.

[⋆] Work currently under submission at another conference.