# Modular and lightweight certification of polyhedral abstract domains*

Alexis Fouilhe, Sylvain Boulmé, and Michaël Périn

Univ. Grenoble Alpes, VERIMAG, F-38000 Grenoble, France
{alexis.fouilhe,sylvain.boulme,michael.perin}@imag.fr

Abstract interpretation [5] provides a theory for static analysis of programs, where sets of reachable states are over-approximated by elements of an *abstract domain*. In particular, the domain of *convex polyhedra* [6] expresses postconditions as conjunctions of affine inequalities: a polyhedron $p$ encodes a formula "$\bigwedge_i \sum_j a_{ij}.x_j \leq b_i$", where $a_{ij}$ and $b_i$ are rational constants, and $x_j$ are numerical variables of the program. Its semantics (or concretization) is the *predicate* $[\![p]\!]$ defined as "$\lambda m. \bigwedge_i \sum_j a_{ij}.m(x_j) \leq b_i$", where $m$ is a total map from variables to rationals representing a *memory state*. The analyzer computes postconditions in a given abstract domain by performing a symbolic evaluation of programs that combines *operators* of this domain. Its correctness relies on each domain operator over-approximating a given *predicate transformer*.

An abstract interpreter such as Astrée [3] is able to ensure the absence of undefined behaviours in large critical programs from avionics. But Astrée is itself very complex and, despite the care put in its development, it may contain bugs. This is probably also the case for well-known abstract domain implementations, such as the PPL [1] and Apron [8]. Inspired by the development in Coq of the CompCert certified compiler [10], the Verasco project aims to build a certified abstract interpreter [4]. Our work in this project focuses on obtaining a provably correct library for convex polyhedra, similar in features and performance to the core of the PPL and Apron polyhedra libraries.

Proving correct the result of domain operators on polyhedra reduces to proving inclusions of polyhedra: a polyhedron $p$ is included in a polyhedron $p'$ iff $\forall m, [\![p]\!](m) \Rightarrow [\![p']\!](m)$. If each inequality of $p'$ is entailed by a positive linear combination of the inequalities of $p$, then inclusion holds. Farkas's lemma states that when inclusion holds, such a vector $\Lambda$ of linear combinations always exists.

Moreover, such a $\Lambda$ can be considered as a certificate containing the necessary information to build the result $p'$ of a given domain operator from its operands which are here expressed as $p$. The result $p' = \Lambda.p$ satisfies the inclusion properties which guarantee its correctness, by construction. Our certified abstract domain of polyhedra is built out of two components:

- An untrusted Ocaml backend which, for each operator, produces certificates.
- A frontend, developed in Coq, which builds proved-correct results using certificates provided by the backend.

This idea has previously been experimented by Frédéric Besson et al. [2]. Our work makes the frontend more modular and more generic with respect to the backend. All that is required from the backend is to be able to generate certificates in our format. The backend could use its own data structures (e.g. double representation), or trade some precision for computationally cheaper operators [11, 9]. Such flexibility is achieved reducing the coupling between the frontend and the backend:

- Communication between the frontend and the backend is reduced to certificates, i.e. descriptions of linear combinations of inequalities identified by integers.

*This work was partially supported by ANR project "Verasco" (INS 2011).

- The frontend ensures soundness but does not give formal precision guarantees. The precision versus efficiency trade-off is delegated to the backend.

The frontend requires the backend to implement only a basic OCaml interface. It is extended in the frontend using functors: extra features are added in a modular way to any numerical domain without relying on its specifics. For example, the predicate transformer for assignment can be phrased in terms of more basic operators: intersection, projection and renaming. A functor adds the operator to a basic domain and builds the required correctness proofs.

To complete our abstract domain, we built a backend using a constraints-only representation of polyhedra [7]. Its operators use tweaked versions of standard algorithms so as to produce certificates as a cheap by-product of computations. Experiments show that the overhead of result verification is sufficiently low for our abstract domain to remain competitive with well-established, but non-verifying, implementations.

In conclusion, result verification is particularly well suited for certifying polyhedral abstract domains. Our work demonstrates an efficient, evolutive and reusable design, which could serve as a guiding example for lightweight certification. We hope to extend this work to a whole static analyzer.

# References


[1] R. Bagnara, P. M. Hill, and E. Zaffanella. The Parma Polyhedra Library: Toward a complete set of numerical abstractions for the analysis and verification of hardware and software systems. *Science of Computer Programming*, 72(1–2), 2008.

[2] F. Besson, T. Jensen, D. Pichardie, and T. Turpin. Result certification for relational program analysis. Technical Report RR-6333, INRIA, 2007.

[3] B. Blanchet, P. Cousot, R. Cousot, J. Feret, L. Mauborgne, A. Miné, D. Monniaux, and X. Rival. A static analyzer for large safety-critical software. In *PLDI*. ACM, 2003.

[4] S. Blazy, V. Laporte, A. Maroneze, and D. Pichardie. Formal Verification of a C Value Analysis Based on Abstract Interpretation. In *SAS*, volume 7935 of *LNCS*. Springer, 2013.

[5] P. Cousot and R. Cousot. Abstract interpretation: a unified lattice model for static analysis of programs by construction or approximation of fixpoints. In *POPL*. ACM, 1977.

[6] P. Cousot and N. Halbwachs. Automatic discovery of linear restraints among variables of a program. In *POPL*. ACM, 1978.

[7] A. Fouilhe, D. Monniaux, and M. Périn. Efficient Generation of Correctness Certificates for the Abstract Domain of Polyhedra. In *SAS*, volume 7935. Springer, 2013.

[8] B. Jeannet and A. Miné. Apron: A library of numerical abstract domains for static analysis. In *CAV*, 2009.

[9] V. Laviron and F. Logozzo. Subpolyhedra: a (more) scalable approach to infer linear inequalities. In *VMCAI*, volume 5403 of *LNCS*, pages 229–244. Springer, 2009.

[10] X. Leroy. Formal verification of a realistic compiler. *Communications of the ACM*, 52(7), 2009.

[11] S. Sankaranarayanan, M. Colón, H. Sipma, and S. Manna. Efficient strongly relational polyhedral analysis. In *VMCAI*, volume 3855 of *LNCS*, pages 111–125. Springer, 2006.