A proof-theoretic approach to abstract interpretation

Apostolos Tzimoulis

joint work with Vijay D'Silva, Alessandra Palmigiano, Caterina Urban

Abstract interpretation is a theory of formal program verification which generates sound approximations of the semantics of programs, and has been used as the basis of methods and effective algorithms to approximate undecidable or computationally intractable problems such as the verification of safety-critical software (e.g. medical, nuclear, aviation software).

Typically, a complex concrete model (such as the powerset $\mathcal{P}(\Sigma)$ of a possibly infinite set modelling program executions) is related to a model that can be efficiently represented and manipulated, which is usually a finite lattice A that encodes the relevant – logically interconnected – properties about these executions, by means of an adjoint pair of maps. Specifically, the right adjoint (the *concretization* map $\gamma : A \to \mathcal{P}(\Sigma)$) provides the intended interpretation of the symbolic properties (that is, $S \models a$ iff $S \subseteq \gamma(a)$ for any $S \in \mathcal{P}(\Sigma)$ and $a \in A$); the left adjoint (the *abstraction* map $\alpha : \mathcal{P}(\Sigma) \to A$) classifies the executions of the given program according to their satisfying the relevant properties. Even though the concrete model is usually a Boolean algebra, the algebraic structure that the abstract lattice A retains depends on the properties preserved by the concretization map γ.

Although this theory was connected to logic since its inception [2, 1, 4], it is only in the last decade that the connection was made systematic. In particular, the notion of an (internal) logic of an abstraction was introduced in [5] and systematically related to the order-theoretic properties of the concretization map. In [3], this line of research is further developed. Namely, the logics underlying specific abstractions are identified, together with explicit specification of proof-theoretic presentations for each of them.

The present talk reports on the results of an ongoing work in which, using duality theory and algebraic logic, we generalise the results of [3] and introduce a general procedure for generating the (internal) logic of an abstraction together with the specification of a proof system for it. The main idea is to generate a logic whose Lindenbaum-Tarski algebra is isomorphic to the abstract algebra A. In particular, we highlight the connection between properties of the logic, such as its expressiveness and its completeness, and the preservation properties of the concretization map. We further discuss issues on the optimality of such logics, as well as combinators of such algebras and the algebraic properties they preserve. Ongoing research directions concern the extension of these results to richer abstract algebras A endowed with modal (dynamic) operators.

References

[1] P. Cousot. Semantic foundations of program analysis. In S. S. Muchnick and N. D. Jones, editors, *Program Flow Analysis: Theory and Applications*, pages 303–342. Prentice Hall, 1981.

[2] P. Cousot. Méthodes itératives de construction et d' approximation de points fixes d' opérateurs monotones sur un treillis, analyse sémantique de programmes. *PhD thesis*, University of Grenoble, 1978.

[3] V. D'Silva and C. Urban. Abstract interpretation as automated deduction. *Journal of Automated Reasoning*, 58(3):363–390, 2017.

[4] T. P. Jensen. Abstract interpretation in logical form. *PhD thesis*, Imperial College, University of London, 1992.

[5] D. A. Schmidt. Internal and external logics of abstract interpretations. In *International Workshop on Verification, Model Checking, and Abstract Interpretation*, pages 263–278. Springer, 2008.