Discrete Fixpoint Approximation Methods in Program Static Analysis

Patrick Cousot

École Normale Supérieure
DMI, 45, rue d'Ulm, 75230 Paris cedex 05, France
cousot@dmi.ens.fr http://www.dmi.ens.fr/~cousot

Keywords: abstract interpretation, static analysis, discrete fixpoints, fixpoint approximation, Galois connexion, chaotic iteration, asynchronous iteration, acceleration of convergence, widening, narrowing.

AMS subject classification: 06-A15, 06-A23, 65-B99, 68-N20, 68-Q55, 68-Q60

Static analysis consists in automatically determining runtime properties of the specification of computer systems, in particular programs, by inspection of the text (excluding executions or simulations). Examples of applications include software engineering (test generation, abstract debugging), compiling (dataflow analysis, type inference), program transformation (vectorization, parallelization).

Abstract interpretation [1] is a design method for static analysis algorithms by effective approximation of the semantics of programs. Most often, the semantics maps the program text to a model of computation obtained as the least fixpoint of an operator on a partially ordered semantic domain.

The idea of simplifying the fixpoint system of semantic equations can be formalized as a fixpoint transfer between the concrete semantics and an abstract/approximate fixpoint semantics formally derived from the concrete one through (among others) a Galois connexion between the partially ordered semantic domains.

The abstract system of fixpoint equations can then be solved iteratively using chaotic or asynchronous iterations. With infinite abstract domains, widening and narrowing operators must be used in order to guarantee the existence of effective approximations, to enforce termination or accelerate convergence.

The formal theory of abstract interpretation is shortly sketched and then exemplified for numerous numerical domains including interval, congruence and polyhedral approximation.

Reference:

[1] P. Cousot and R. Cousot. Abstract interpretation frameworks. *Journal of Logic and Computation*, 2(4):511–547, August 1992.