Numerical Abstract Domains for Digital Filters

Jérôme Feret

Numerical Abstract Domains for Digital Filters*

Jérôme Feret

DI, École Normale Supérieure, Paris, FRANCE
jerome.feret@ens.fr

Abstract We design specific domains for analyzing digital filtering. Digital filtering consists in implementing numerical recursions: the value of a variable S (the output) is computed from a fixed finite number of the last consecutive values of the variable S and from a fixed finite number of the last consecutive values of another variable E (the input). Our framework allows us to refine existing analyses so that they can handle given classes of digital filters. We only have to design a class of symbolic properties that describe the invariants throughout filter iterations, and to describe how these properties are transformed by filter iterations. Then, the analysis allows both inference and proofs of the properties about the program variables that are tied to any such filter. In case of linear filters, we propose a systematic method for designing the abstract domain by using intervals and ellipsoidal constraints.

1 Designing a numerical domain

An abstract domain provides all the elementary tools for dealing with a given class of properties. Given a specific issue, we build the set of all the constraints that may be required for proving this kind of properties. This choice is the first approximation we make. For the sake of efficiency, we may choose not to express some properties in this set of constraints, this is the second source of approximation. Then, we have to define sound counterparts to the primitives of the concrete semantics. These counterparts may be accurate enough not to lose too much information and efficient enough to scale up: this is the third source of approximation. Moreover, in the case of a relational domain, soundness may require that computations are performed in the real field (since domains usually combine their constraints by using commutativity and associativity properties). Scalability can be obtained by over-approximating these computations by floating-point numbers, which is another cause of approximation. For the sake of expressiveness, domains are usually not height-bounded. Property inference then requires extrapolation operators, this is the fourth source of approximation. Lastly, since a complex analysis usually yields several classes of properties, we may have to transfer information between several abstract domains. The most accurate reduced product may not be decidable, or too much expensive. We have to choose an efficient way to perform an approximation of the reduced product.

* This work was partially supported by the ASTRÉE RNTL project.

2 Digital filter domain

We meet all these difficulties when designing abstract domains for analyzing digital filters. Given a generic form of recursive sequence (such as $S_{n+2} = aS_{n+1} + bS_n + cE_{n+2} + dE_{n+1} + eE_n$) computed in the floating-point world, an abstract property relates some variables (here S_{n+2}, S_{n+1}, E_{n+2}, and E_{n+1}) to some abstract values. Such a property expresses that, up to rounding errors, the variables S_{n+2} and S_{n+1} are associated with two consecutive values of the recursive sequence while the variables E_{n+2} and E_{n+1} are associated with the last two input values. The abstract values capture the input stream (E_n), initial conditions S_0 and S_1 and the overall contribution of rounding errors. We give a framework to deal with filter iteration, filter reinitialization, branching, loop, and so on, provided that the analysis designer provides some transfer functions for filter iteration. In the case when the recursion is linear (up to rounding errors) the analysis designer has only to provide a generic form of invariants for dealing with simplified recursion in the real field, transfer functions for this invariant, and conversion between these kinds of constraints and interval constraints. In our case, the simplified recursion is of the form $S_{n+2} = a.S_{n+1} + b.S_n + F_n$, invariants are of the form $S_{n+2}^2 - aS_{n+1}S_n - bS_n^2 \leq K$, and transfer functions are given in [2]. Most of the difficulties lie in the approximated reduced product with the already existing abstract domains. Reduction steps are performed when necessary not to lose accuracy. Intervals and equality constraints can be collected on the fly to capture initial conditions. The most difficult part lies in refining interval constraints using filter constraints.

3 Bounding linear filter output

We give a systematic method when the recursion is linear.

3.1 Formal expansion and rounding errors

Formally, the output can be split into three summands: the contribution of the last N inputs if the digital filter were computed in the real field, the contribution of the initial outputs and — of the other inputs if the digital filter were computed in the real field, and the difference between the computation in the floating-point world and in the real field. The integer N is a parameter of the approximation. The first summand can be symbolically computed as a known function from the last input values into the real field. The second and the third summands both satisfy the simplified recursion which allows the easy computation of a sound bound for them. Relative rounding errors at each filter iteration can be bounded by using [7].

3.2 Linear filters in any dimension

We can bound the output of simplified linear filters by splitting the simplified recursion as a sum of second order recursions and of first order recursions. We

require a factorization of the linear recursion characteristic polynomial into irreducible (in the real field) polynomials (we only need a sound approximation of the coefficient of each polynomial). We also need that the characteristic polynomial has only single roots. Then, we can bound second order summands by using ellipsoidal constraints [2], while first order summands can be bounded by using interval constraints [3]. These bounds can be discovered iteratively by using simple transfer function. We have given in [4] necessary conditions (taking into account rounding errors) for proving the convergence of first and second order filters, and we have given explicit bounds that can be used to accelerate abstract iterations. Nevertheless only the soundness of the usual transfer function is required to prove the soundness of the analysis [4]: the analysis is sound even if the acceleration is not sound. In the case when these necessary conditions are not satisfied, we can use the arithmetic geometric progression domain [5] in order to compute bounds that depend of the program life time.

4 Conclusion

Our framework allows proving the absence of arithmetic overflows due to the use of digital filters in industrial software. The abstract domain for dealing with first order and second order stable filters have been implemented in the ASTRÉE analyzer [1,6]. These domains were required for certifying the absence of run-time errors in huge industrial application (ranging from $75,000$ to $379,000$ lines of C). Moreover, we believe that our approach can provide insightful ideas for designing new specific numerical domains.

References

1. Astrée. static analysis of critical real-time embedded software, analyzer page. http://www.astree.ens.fr/.
2. 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 *Proc. PLDI'03*. ACM Press, 2003.
3. P. Cousot and R. Cousot. Static determination of dynamic properties of programs. In *Proceedings of the Second International Symposium on Programming*, pages 106–130. Dunod, Paris, France, 1976.
4. J. Feret. Static analysis of digital filters. In *European Symposium on Programming (ESOP'04)*, number 2986 in LNCS. Springer-Verlag, 2004.
5. J. Feret. The arithmetic-geometric progression abstract domain. In *Verification, Model Checking and Abstract Interpretation (VMCAI'05)*, number 3385 in LNCS, pages 42–58. Springer-Verlag, 2005.
6. L. Mauborgne. ASTRÉE: Verification of absence of run-time error. In René Jacquart, editor, *Building the information Society (18th IFIP World Computer Congress)*, pages 384–392. The International Federation for Information Processing, Kluwer Academic Publishers, Aug 2004.
7. A. Miné. Relational abstract domains for the detection of floating-point run-time errors. In *Proc. ESOP'04*, LNCS. Springer, 2004.