Using Gröbner basis theory for an interval method solving underdetermined equations systems

Bartłomiej Jacek Kubica[1]

[1] *Department of Applied Informatics, Warsaw University of Life Sciences, Poland, bartlomiej_kubica@sggw.pl*

Let us consider solving the nonlinear underdetermined system of equations:

$$f\colon X \to \mathbb{R}^m, \text{ where } X \subseteq \mathbb{R}^n,\ n \geq m\ . \tag{1}$$

Interval methods (see, e.g., [6]) have proven to be useful, in particular, in solving nonlinear systems of type (1). One of their advantages is allowing not only to locate all solutions of underdetermined systems; i.e., the whole solution manifold can be enclosed by a set of boxes (typically we compute two sets: of verified and possible solutions, cf., e.g., [9]).

Due to the nature of interval arithmetic, it is pretty important, what formulae we compute in it. The simplest example is $[\underline{x}, \overline{x}] - [\underline{x}, \overline{x}]$, which, according to the rules of interval arithmetic is equal to $[\underline{x} - \overline{x}, \overline{x} - \underline{x}]$ and it is in general different from zero.

It might be unlikely that we found a $\boldsymbol{x} - \boldsymbol{x}$ in our formulae, but also $\boldsymbol{x}^2 + \boldsymbol{x}$, $\boldsymbol{x} \cdot (\boldsymbol{x} + 1)$ and $(\boldsymbol{x} + \frac{1}{2})^2 - \frac{1}{2}$, obviously equivalent for real numbers, may have different results for an interval argument.

Hence, combining interval methods with some symbolic transformations might be very worthwhile.

Benhamou et alii were, to the best knowledge of the author, the first ones to propose preprocessing equations systems under consideration using the Gröbner basis theory [2], [3]. Computing the Gröbner basis of a set of polynomials, corresponding to the equation system, in lexicographic order $x_1 \prec x_2 \prec \cdots \prec x_n$, results in a system in triangular form:

$$\begin{cases} p_1(x_1, x_2, \ldots, x_n) = 0 \\ \cdots \\ p_{n-1}(x_1, x_2) = 0 \\ p_n(x_1) = 0 \end{cases} .$$

Obviously, variables in the above ordering can be permuted, resulting in a different transformed system, but also in a triangular form.

The transformation thus allows us to reduce solving the whole system to subsequent solving of univariate equations: $p_n(x_1) = 0$, $p_2(x_1, x_2) = 0$, for solutions x_1^* of the previous equation, etc. The procedure, according to the quoted papers is efficient. A similar idea has been applied by the author for solving optimization problems; see [7].

In all above cases, the system of transformed conditions gets reduced to the triangular form. It is not so for an underdetermined system of equations, where we only get the following transformed system:

$$\begin{cases} p_1(x_1, \ldots, x_{n-m+1}, \ldots, x_n) = 0, \\ \ldots \\ p_{m-1}(x_1, \ldots, x_{n-m+1}, x_{n-m+2}) = 0, \\ p_m(x_1, \ldots, x_{n-m+1}) = 0 \end{cases} .$$

Here, we need to start with solving a multivariate underdetermined equation $p_m(x_1, \ldots, x_{n-m+1}) = 0$. Let us denote the solution manifold of this equation $M = \{(x_1, \ldots, x_{n-m+1}) \mid p_m(x_1, \ldots, x_{n-m+1}) = 0\}$. We obtain M as a set of boxes enclosing its segments (cf., e.g., [9]).

For all these boxes, we can proceed with solving univariate equations to find the solution of the initial system (1), as in the well-determined case.

Computing M is obviously, much more demanding and cumbersome than solving a univariate equation, but still it is an improvement: instead of solving a system of m equations in n variables, we need to enclose the solution manifold of a single equation in $(n - m + 1)$ variables.

What is more, next steps, in which we compute feasible values of x_{n-m+2}, x_{n-m+3}, ..., x_n can be parallelized in a pretty scalable manner: M is probably enclosed by a large number of boxes and computations for each of these boxes are independent on computations on the others.

To the best knowledge of the author, this approach has not been considered or tested for underdetermined systems of equations and this paper is going to fill this gap.

The solver used by the author is HIBA_USNE [5], written by himself and described, i.a., in [9], [10]. For symbolic preprocessing, CoCoALib [1] is applied.

References

[1] J. Abbott and A. M. Bigatti, *CoCoALib: a C++ library for doing Computations in Commutative Algebra*, `http://cocoa.dima.unige.it/cocoalib` (2017).

[2] F. Benhamou and L. Granvilliers, *Combining local consistency, symbolic rewriting and interval methods*, Artificial Intelligence and Symbolic Mathematical Computation, **1996**, pp. 144–159 (1996), `http://www.sciences.univ-nantes.fr/info/perso/permanents/benhamou/papers/BenGra_AISMC96.pdf`.

[3] F. Benhamou and L. Granvilliers, *Automatic generation of numerical redundancies for non-linear constraint solving*, Reliable Computing, **3**, 3, pp. 335–344 (1997), http://www.sciences.univ-nantes.fr/info/perso/permanents/benhamou/papers/BenGra_Reliable97.pdf.

[4] B. Buchberger, *Gröbner bases and systems theory*, Multidimensional Systems and Signal Processing, *12*, 3, pp. 223–251 (2001), http://www.risc.uni-linz.ac.at/people/buchberg/papers/2001-05-12-A.ps.

[5] *HIBA_USNE*, C++ library, https://www.researchgate.net/publication/316687827_HIBA_USNE_Heuristical_Interval_Branch-and-prune_Algorithm_for_Underdetermined_and_well-determined_Systems_of_Nonlinear_Equations_-_Beta_25 (2017).

[6] R. B. Kearfott, *Rigorous Global Search: Continuous Problems*, Kluwer, Dordrecht, 1996.

[7] B. J. Kubica and K. Malinowski, *An interval global optimization algorithm combining symbolic rewriting and componentwise Newton method applied to control a class of queueing systems*, Reliable Computing, **11**, 5, pp. 393–411 (2005).

[8] B.J. Kubica, *A class of problems that can be solved using interval algorithms*, Computing, **94**, pp. 271–280 (2012).

[9] B.J. Kubica, *Presentation of a highly tuned multithreaded interval solver for underdetermined and well-determined nonlinear systems*, Numerical Algorithms, **70**, 4, pp. 929–963 (2015).

[10] B.J. Kubica, *Parallelization of a bound-consistency enforcing procedure and its application in solving nonlinear systems*, Journal of Parallel and Distributed Computing, published online https://doi.org/10.1016/j.jpdc.2017.03.009 (2017).