

About avoidable computations in interval methods

Narendra JUSSIEN, Olivier LHOMME

École des Mines de Nantes

La Chantrerie - 4, rue Alfred Kastler - B.P. 20722, 44 307 Nantes Cedex 3, France

E-mail: {Olivier.Lhomme}@emn.fr

Abstract

Interval methods for solving systems of nonlinear equations typically split an interval vector or a box each time the Newton operator $N(x, X)$ does not lead to a sufficient narrowing. The drawback of such a splitting procedure is that, when applied recursively, it may generate many sub-boxes, and for each sub-box, it is necessary to apply the Newton operator from scratch: the Jacobian and its inverse have to be computed, what is computationally expensive. So, an exponential number of expensive computations must be done. The behavior of the overall Interval Newton method is generally much better, because the Newton operator $N(x, X)$ generally performs well, so the number of splitting is low. But in some very hard cases (see for example the problem in [3]) a great number of sub-boxes are really generated.

Two questions come to mind:

- How can the number of generated sub-boxes be decreased?
- How can the computations over a sub-box be reused for another sub-box?

We will address in this paper both questions.

The idea is to maintain a dependency graph that allows to keep reusable information when leaving a sub-box I_1 for another one I_2 . When the information which is kept is that the sub-box I_2 cannot contain any zero it is easy to see that this will avoid I_2 to be tried, so possible sub-boxes of I_2 are not generated.

This work is inspired by works done in Artificial Intelligence: especially that of Ginsberg [1] and Jussien and Lhomme [2].

References

1. Matthew L. Ginsberg, Dynamic Backtracking, *Journal of Artificial Intelligence Research* volume 1, pages 25–46, 1993.
2. N. Jussien, O. Lhomme, *Dynamic Domain Splitting*, Proceedings of ECAI-98. Forthcoming, 1998.
3. Van-Hentenryck, P., Puget, J-F., *A Constraint Satisfaction Approach to a Circuit Design Problem*. Journal of Global Optimization (Forthcoming).