Filtering Domains of Factorable Functions using Interval Contractors

Laurent Granvilliers

Filtering Domains of Factorable Functions using Interval Contractors

Laurent Granvilliers

LS2N, Université de Nantes, France
laurent.granvilliers@univ-nantes.fr

Abstract. Many theorems in mathematics require a real function to be continuous over the domain under consideration. In particular the Brouwer fixed point theorem and the mean value theorem underlie many interval methods like Newton operators for solving numerical constraint satisfaction problems or global optimization problems. Since the continuity property collapses when the function is not defined at some point it is then important to check whether the function is defined everywhere in a given domain. We introduce here an interval branch-and-contract algorithm that rigorously approximate the domain of definition of a factorable function within a box. The proposed approach mainly relies on interval contractors applied to the domain constraints and their negations stemming from the expression of the function.

Keywords: Interval methods · Branch-and-contract algorithm · Interval contractor · Constraint satisfaction problem · Paving.

1 Introduction

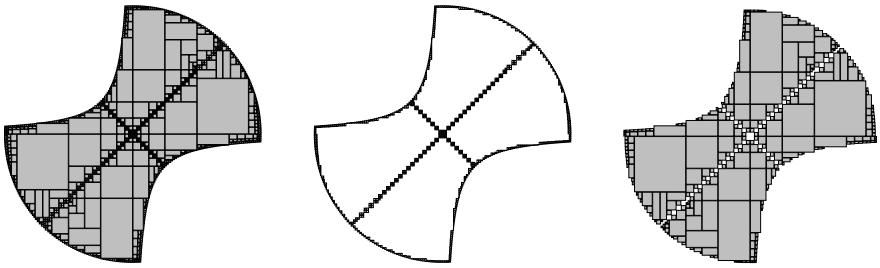
A real function $f : D \to \mathbb{R}^m$ with $D \subseteq \mathbb{R}^n$ is *factorable* if it can be defined as a finite recursive composition of arithmetic operations and elementary functions simply called operations thereafter. Given a box $\Omega \subseteq \mathbb{R}^n$ we study the problem of approximating the intersection $\Omega \cap D$ with interval computations. Our goal is to calculate a paving $(\mathcal{X}^i, \mathcal{X}^o)$ where $\mathcal{X}^i$ is a union of inner boxes and $\mathcal{X}^o$ is a union of outer boxes such that

$$\mathcal{X}^i \subseteq \Omega \cap D \subseteq \mathcal{X}^i \cup \mathcal{X}^o \subseteq \Omega.$$

Fig. 1 shows such a paving computed at a given precision $\epsilon > 0$. We see that the outer boxes are accumulated on the frontier of the set $\Omega \cap D$, the width of each one defined componentwise being smaller than ϵ.

The problem described above can be defined as a numerical constraint satisfaction problem (CSP) $\langle \mathcal{C}, \Omega \rangle$ where $\mathcal{C}$ is a set of constraints such that a point $x \in \Omega$ belongs to D if and only if it satisfies all the constraints from $\mathcal{C}$. It turns out that every operation of f having a restricted domain entails a constraint that must be inserted in $\mathcal{C}$. For example, the function whose paving is depicted in Fig. 1 leads to the set

$$\mathcal{C} = \{x_1 x_2 + 1 \geq 0, x_1^2 - x_2^2 \neq 0, 16 - x_1^2 - x_2^2 > 0\}.$$



(a) (b) (c)

Fig. 1. Let $f(x_1, x_2) = (\sqrt{x_1 x_2 + 1}/(x_1^2 - x_2^2), \log(16 - x_1^2 - x_2^2))$ be a real function. Fig. (*a*) shows a paving of its domain of definition at precision $\epsilon = 0.1$ given $\Omega = \mathbb{R}^2$ composed of outer boxes depicted in Fig. (*b*) and inner boxes depicted in Fig. (*c*).

We see that the frontier of its domain of definition is delimited by an hyperbola, a circle and lines represented by the domain constraints.

A paving of the solution set of a numerical CSP can be computed by an interval branch-and-contract algorithm that recursively divides and contracts the initial box until reaching the desired precision, following a contractor programming approach [5]. In this framework, an interval contractor is associated with one or many constraints to tighten a box by removing inconsistent values from its bounds, using different techniques such as consistency techniques adapted to numerical computations with intervals [11] or interval Newton operators [10]. Several contractors can be applied in a fixed-point loop known as constraint propagation [12]. Finding inner boxes can be done by considering the negations of the constraints [2] or by means of inflation techniques [6, 4].

In the following, we introduce an interval branch-and-contract algorithm that calculates a paving of the domain of definition of a real function within a given box. A set of rules is proposed to derive the system of domain constraints entailed by the expression of the function. We adapt the HC4Revise contractor [3] in order to process these specific constraints. Finally, a new heuristic for generating maximal inner boxes is devised. A set of experiments permit to evaluate the quality of computed pavings.

The rest of this paper is organized as follows. Interval arithmetic and the notion of interval contractor will be introduced in Section 2. The new algorithms will be described in Section 3. Section 4 is devoted to the experimental results, followed by a conclusion.

2 Interval computations

2.1 Interval arithmetic

An interval is a closed and connected set of real numbers. The set of intervals is denoted by $\mathbb{I}$. The empty interval represents an empty set of real numbers.

The width of an interval $[a, b]$ is equal to $(b - a)$. The interval hull of a set of real numbers S is the interval $[\inf S, \sup S]$ denoted by $\operatorname{hull} S$. Given an integer $n \geq 1$, an n-dimensional box X is a Cartesian product of intervals $X_1 \times \cdots \times X_n$. A box is empty if one of its components is empty. The width of a box X is the maximum width taken componentwise denoted by $\operatorname{wid} X$.

Interval arithmetic is a set extension of real arithmetic [13]. Let $g : D \to \mathbb{R}$ be a real function with $D \subseteq \mathbb{R}^n$. An interval extension of g is an interval function $G : \mathbb{I}^n \to \mathbb{I}$ such that

$$(\forall X \in \mathbb{I}^n)\ (\forall x \in X \cap D)\ g(x) \in G(X).$$

This property called the *fundamental theorem of interval arithmetic* implies that the interval $G(X)$ encloses the range of g over X. When g corresponds to a basic operation, it is possible to implement the interval operation in a way to calculate the hull of the range by exploiting monotonicity properties, limits and extrema. More complex functions can be extended in several ways. In particular, the natural interval extension of a factorable function consists of evaluating the function with interval operations given interval arguments.

2.2 Interval contractors

Given a vector of unknowns $x \in \mathbb{R}^n$, an *interval contractor* associated with a constraint $c(x)$ is an operator $\Gamma : \mathbb{I}^n \to \mathbb{I}^n$ verifying the following properties:

$$(\forall X \in \mathbb{I}^n) \quad \begin{cases} \Gamma(X) \supseteq \{x \in X : c(x)\} & \text{(consistency)} \\ \Gamma(X) \subseteq X & \text{(contractance)} \end{cases}$$

An interval contractor aims at removing inconsistent values at the bounds of the variable domains. There are many kinds of contractors and we present here the forward backward contraction algorithm called HC4Revise [3]. Given an equation $g(x) = 0$ or an inequality constraint $g(x) \leq 0$ and a box X, the first phase is an evaluation of the natural extension of g from the leaves to the root. We then consider the interval I associated with the relation symbol, namely $[0, 0]$ for an equation and $[-\infty, 0]$ for an inequality. There are three cases: if the intersection $G(X) \cap I$ is empty then the constraint is inconsistent; if we have the inclusion $G(X) \subseteq I$ then the constraint is consistent and X is an inner box for this constraint said to be inactive; otherwise the second phase calculates projections from the root of g lying in $G(X) \cap I$ to the leaves, eventually contracting the variable domains. An example is presented in Fig 2.

As previously shown, an HC4Revise contractor is able to detect that a box is an inner box after the first phase. Now it is possible to apply it to the negation of a constraint in order to generate inner boxes inside a box, as follows. Given an inequality constraint $g(x) \leq 0$, let Γ be an HC4Revise contractor associated with its negation $g(x) > 0$. Given a box X, it comes by the consistency property of Γ that every element of the region $X \setminus \Gamma(X)$ violates the constraint negation, hence satisfying the constraint itself. When this region is non empty, it is possible to generate inner boxes for the constraint, as shown in Fig. 3. Since the rounding

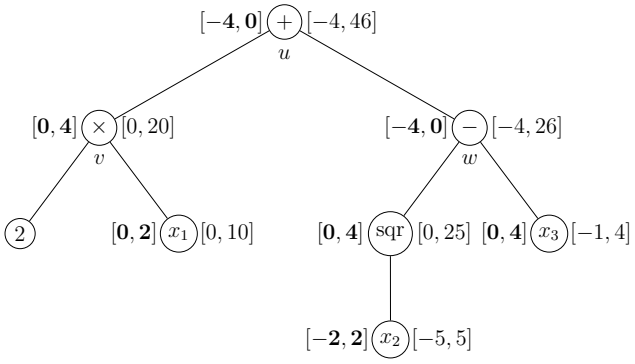


Fig. 2. Let $g(x) \leq 0$ be an inequality constraint with $g(x_1, x_2, x_3) = 2x_1 + x_2^2 - x_3$ and let X be the box $[0, 10] \times [-5, 5] \times [-1, 4]$. The interval on the right at each node of g is the result of the interval evaluation phase of the HC4Revise contractor. The interval at the root node is intersected with the interval $I = [-\infty, 0]$ associated with the relation symbol. The interval on the left at each node is the result of the projection phase from the root to the leaves. For example, let $u \in [-4, 0]$, $v \in [0, 20]$ and $w \in [-4, 26]$ be three variables respectively labelling the $+$ node, the $\times$ node and the $-$ node. We have to project the equation $v + w = u$ over v and w, which propagates the new domain at the root node to its children nodes. To this end the equation is inverted and it is equivalently rewritten as $v = u - w$. The new domain for v is calculated as $[0, 20] \cap ([-4, 0] - [-4, 26])$, which leads to the new domain $[0, 4]$ at the $\times$ node. The new domain for w is derived similarly. At the end of this backward phase it comes the new box $[0, 2] \times [-2, 2] \times [0, 4]$.

errors of machine computations prevent in general to deal with open intervals, the constraint negation is safely relaxed as $g(x) \geq 0$.

3 Filtering Domains of Functions

3.1 Domain constraints

Several operations have restricted domains such as the division $x \mapsto x^{-1}$ defined in $\mathbb{R} \setminus \{0\}$, the square root defined in $\mathbb{R}^+$, the logarithm defined in $(0, +\infty)$, the arccosine and arcsine functions defined in $[-1, 1]$ and the tangent function defined at every point that is not a multiple of $\pi/2$. A factorable function whose definition involves one of these operations may not be defined everywhere in a box, and, *a fortiori*, it may not be continuous. It naturally yields domain constraints that must be verified, as illustrated by Fig. 4.

Every term $op(u_1, \ldots, u_k)$ occurring in a factorable real function $f : D \subseteq \mathbb{R}^n \to \mathbb{R}^m$ such that the domain of op is a strict subset of $\mathbb{R}^k$ entails a constraint. A constraint system $\mathcal{C}$ can then be generated from f using the following (non exhaustive) set of rules. There are different kinds of constraints such as

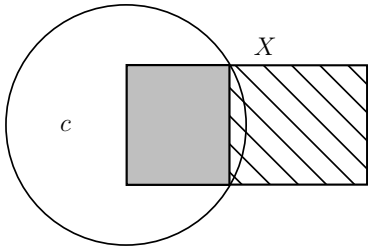


Fig. 3. Let c be the inequality constraint $x_1^2 + x_2^2 \leq 4$ that defines a disk in the Euclidean plane and let X be the box $[0, 4] \times [-1, 1]$. The hatched surface is returned by an HC4Revise contractor Γ associated with the negation of c. The gray region $X \setminus \Gamma(X)$ is thus an inner region for c (every point satisfies c) and it is a box here.

(strict and non-strict) inequality constraints and disequations. The algorithms introduced thereafter will also consider their negations.

$$\begin{cases} \sqrt{u} & \models & u \geq 0 \\ \log u & \models & u > 0 \\ 1/u & \models & u \neq 0 \\ \operatorname{acos} u & \models & -1 \leq u \leq 1 \\ \operatorname{asin} u & \models & -1 \leq u \leq 1 \\ \tan u & \models & u \neq \pi/2 + k\pi \ (k \in \mathbb{Z}) \end{cases}$$

Given a box $\Omega \subseteq \mathbb{R}^n$, every $x \in \Omega$ satisfying all the constraints from $\mathcal{C}$ must belong to D. Finding the set $\Omega \cap D$ is then equivalent to solve the numerical CSP $\langle \mathcal{C}, \Omega \rangle$. It is worth noting that the set $\mathcal{C}$ may be separable. For example, the function $f(x_1, x_2) = \log(x_1) + x_2^{-1}$ entails two constraints $x_1 > 0$ and $x_2 \neq 0$ sharing no variable and thus handable separately.

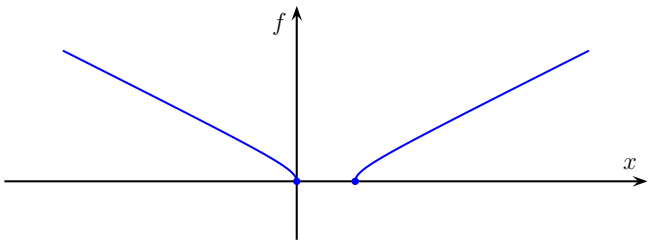


Fig. 4. The function $f(x) = \sqrt{x^2 - x}$ is undefined in the open interval $(0, 1)$ since the square root is defined in $\mathbb{R}^+$ and $g(x) = x^2 - x$ is negative for all x such that $0 < x < 1$. The restricted domain of the square root entails the constraint $x^2 - x \geq 0$.

3.2 Branch-and-contract-algorithm

Algorithm 1 implements a classical interval branch-and-contract algorithm that calculates a paving of the domain of definition of a function f within a given box Ω. It maintains a list $\mathcal{L}$ from which a CSP $\langle C, X\rangle$ is extracted at each iteration. This CSP is reduced and divided by two algorithms *contract* and *branch* that are specific to our problem. If the set C becomes empty then X is inserted in the set of inner boxes $\mathcal{X}^i$. A tolerance $\epsilon > 0$ permits to stop processing too small boxes that are inserted in the set of outer boxes $\mathcal{X}^o$.

Algorithm 1 Branch-and-contract algorithm.

Input: – a function $f : D \to \mathbb{R}^m$ with $D \subseteq \mathbb{R}^n$
– a box $\Omega \subseteq \mathbb{R}^n$
– a tolerance $\epsilon > 0$
Output: – a paving $(\mathcal{X}^i, \mathcal{X}^o)$ of $\Omega \cap D$ at tolerance ϵ
Algorithm:

```
generate the set of domain constraints C from f
initialize L with the CSP ⟨C, Ω⟩
assign (X^i, X^o) to the empty paving
while L is not empty do
   extract an element ⟨C, X⟩ from L
   contract ⟨C, X⟩
   if X ≠ ∅ then
      if C = ∅ then              insert X in X^i
      elif wid X ≤ ε then        insert X in X^o
      else                       branch ⟨C, X⟩
      endif
   endif
endwhile
```

Given a CSP $\langle C, X\rangle$ a contractor is associated with each constraint from the set C. The *contract* component classically implements a constraint propagation algorithm that applies the contractors to reduce X until reaching a fixed-point. Moreover, every constraint detected as inactive is removed from C. The HC4Revise contractor has been designed to handle non-strict inequality constraints and equations since it is not possible to manage open intervals in general due to the rounding errors. The more specific domain constraints are handled as follows. Let $g(x)$ be a real function, let G be the natural interval extension of g and let X be a box.

- A strict inequality constraint $g(x) > 0$ is safely relaxed as $g(x) \geq 0$ since every point that violates the relaxation also violates the constraint.
- A disequation $g(x) \neq 0$ is violated if the interval $G(X)$ is reduced to 0, it is inactive if we have $\max G(X) < 0$ or $\min G(X) > 0$, and nothing happens in the backward phase otherwise.

- A double inequality constraint $a \leq g(x) \leq b$ is simply handled by setting the interval $G(X) \cap [a, b]$ at the root node after the first phase.
- The periodic domain constraint $g(x) \neq \pi/2 + k\pi$ for some integer k does not permit to contract X. The constraint is simply detected as inactive if $G(X)$ does not contain $\pi/2 + k\pi$ for every k, and nothing happens otherwise.

The *branch* algorithm divides a CSP $\langle C, X \rangle$ into sub-problems. Let C be the set of constraints $\{c_1, \ldots, c_p\}$. A contractor Γ_i is associated with the negation of c_i for $i = 1, \ldots, p$. Each contractor is applied to X and it follows that the region

$$X \setminus \bigcup_{i=1}^{p} \Gamma_i(X)$$

is an inner region for the CSP, which means that every point of this region satisfies all the constraints from C, as illustrated in Fig 5.

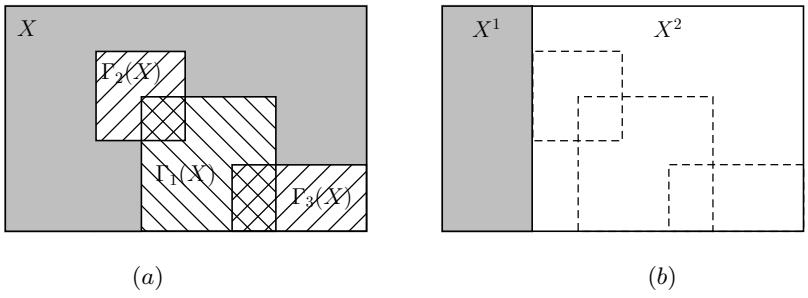


Fig. 5. A box X is contracted by three contractors $\Gamma_1, \Gamma_2, \Gamma_3$ associated with constraint negations, leading to the hatched boxes in Fig. (a). The complementary gray region is an inner region for the original constraints. Fig. (b) shows that X can be split as two boxes $X^1 \cup X^2$ where X^1 is the largest inner slice at one bound of X.

We then define the branching heuristic as follows. Let the box

$$H = \text{hull}\left(\bigcup_{i=1}^{p} \Gamma_i(X)\right)$$

be the interval hull of the contracted boxes with respect to the constraint negations. If H is empty then X is an inner box and it is inserted in $\mathcal{X}^i$. Now suppose that H is not empty. Let $d_i^- = \min H_i - \min X_i$ and $d_i^+ = \max X_i - \max H_i$ be the inter-bound distances between X and H for $i = 1, \ldots, n$. Let

$$d = \max\{d_1^-, \ldots, d_n^-, d_1^+, \ldots, d_n^+\}$$

be the maximum inter-bound distance. If d is greater than the tolerance ϵ then there exists an inner box at one bound of X that is large enough. Assuming for

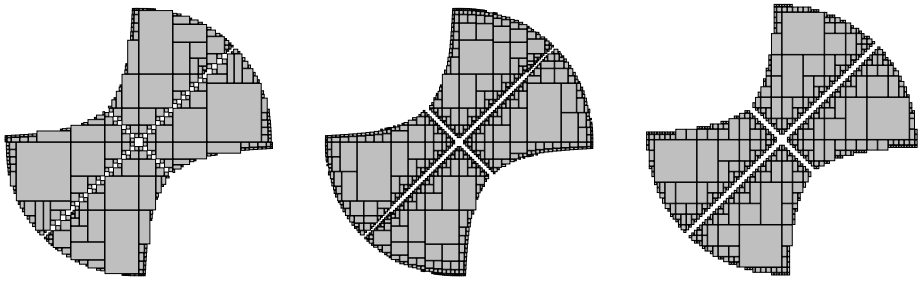
instance that $d = d_j^-$ for some j, X is split in two sub-boxes $X^i \cup X^o$ at $x_j = d_j^-$. The maximal inner box X^i is directly inserted in the set of inner boxes $\mathcal{X}^i$ and the CSP $\langle C, X^o \rangle$ is inserted in $\mathcal{L}$. Otherwise, a bisection of the largest component of X generates two sub-boxes $X' \cup X''$ and the CSPs $\langle C, X' \rangle$ and $\langle C, X'' \rangle$ are added to $\mathcal{L}$, which ensures the convergence of the branch-and-contract algorithm.

4 Experimental results

The interval branch-and-contract algorithm has been developed in the interval solver Realpaver [9]. The interval operations are safely implemented with an outward rounding mode, the MPFR library [7] providing the elementary functions with correct rounding. As a consequence, the interval computations in Realpaver are rigorous. All experiments were conducted on a 64 bits Intel Core i7 4910MQ 2.90GHz processor.

Three strategies will be compared in the following: S_3 corresponds to Algorithm 1, S_2 mimics S_3 but the *split* component always bisects the largest component (no inner box is computed) and S_1 corresponds to S_2 except that the backward phase of the HC4Revise contractors is disabled in the *contract* component (only a satisfaction test is done). The quality of a paving can be measured by its cardinality $(\#\mathcal{X}^i, \#\mathcal{X}^o)$ and the volume of $\mathcal{X}^i$.

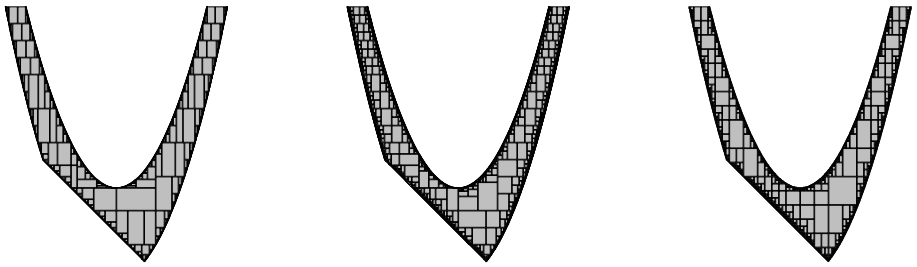
The introductory problem has been processed by S_3, S_2 and S_1 given $\epsilon = 0.1$ and we respectively obtain pavings with cardinalities $(330, 646)$, $(738, 736)$ and $(570, 696)$. There are about the same number of outer boxes which are required to enclose the frontier of the domain of definition at tolerance ϵ. However, the sets of inner boxes depicted in Fig. 6 are much different. S_3 is able to calculate a small number of maximal inner boxes as compared to S_2 and S_1. S_1 generates a regular paving (a quadtree of boxes here). S_2 is able to contract every box, which leads here to increase the number of inner boxes.



(S_3) (S_2) (S_1)

Fig. 6. The sets of inner boxes $\mathcal{X}^i$ computed by the three strategies for the introductory problem at tolerance $\epsilon = 0.1$: 330 boxes for S_1, 738 for S_2 and 570 for S_3. Their total areas are respectively equal to 30.38, 29.95 and 29.74.

Another function involving an arccosine, a square root and a division has been handled and the pavings computed by the three strategies at precision $\epsilon = 0.01$ are depicted in Fig. 7. Their cardinalities are respectively equal to $(1147, 3374)$, $(3187, 3558)$ and $(1896, 2871)$. The surfaces of the sets of inner boxes are respectively equal to 6.962, 6.933 and 6.918. Once again, S_3 generates the best paving with only 1147 inner boxes covering a total area equal to 6.962. S_2 derives a paving with too many boxes as compared to the other strategies but the area covered by the inner boxes 6.933 is slightly better than the one obtained from S_1 equal to 6.918.



(S_3) (S_2) (S_1)

Fig. 7. Given the real function $f(x_1, x_2) = \text{acos}(x_2 - x_1^2) + 1/\sqrt{x_1 + x_2}$ and the box $\Omega = [-5, 5]^2$ the figures above depict the pavings obtained from S_3, S_2 and S_1 using the interval branch-and-contract algorithm applied to the CSP $\langle \mathcal{C}, \Omega \rangle$ given the set of domain constraints $\mathcal{C} = \{-1 \leq x_2 - x_1^2 \leq 1, x_1 + x_2 > 0\}$ generated from f.

These experiments suggest that combining the detection of maximal inner boxes with branching is efficient. On the one hand, this strategy leads to maximize the volume of the set of inner boxes. On the other hand, no more than two sub-boxes are generated at each branching step, which tends to minimize the number of boxes explored during the search.

5 Discussion and perspectives

We have presented an interval branch-and-contract algorithm that rigorously calculates a paving of the domain of definition of a factorable real function. An inner box is a guarantee for interval tests and interval operators that require the continuity property, as motivated in [8] in the context of bound-constrained global optimization. For an inclusion in other methods, it could be interesting to extract from our work a *domain contractor* that returns a union of an inner box included in the domain of definition of the function and an outer box.

The problem studied in this paper has been taken into account by the recent IEEE 1788 standard for interval arithmetic [1]. This standard proposes to

decorate the intervals with different flags including a *dac* flag ensuring that an operation is defined and continuous on the given domain. Implementing a solver on top of an IEEE 1788 compliant interval arithmetic library could then be useful to assert that the result of an interval evaluation has the required property.

In the future, we plan to experiment several inflation techniques [6, 4] and to compare them with the currently implemented method based on the constraint negations. It could be interesting to investigate other branching heuristics and to associate suitable interval contractors with the domain constraints, for instance contractors enforcing strong consistency techniques when those constraints are complex with many occurrences of variables.

Acknowledgment

The author would like to thank Christophe Jermann for interesting discussions about these topics and his careful reading of a preliminary version of this paper.

References

1. 1788-2015, I.S.: IEEE Standard for Interval Arithmetic (2015)
2. Benhamou, F., Goualard, F.: Universally quantified interval constraints. In: Proc. Int. Conf. Principles and Practice of Constraint Progr. (CP). pp. 67–82 (2000)
3. Benhamou, F., Goualard, F., Granvilliers, L., Puget, J.F.: Revising hull and box consistency. In: Proc. Int. Conf. Logic Progr. (ICLP). pp. 230–244 (1999)
4. Chabert, G., Beldiceanu, N.: Sweeping with continuous domains. In: Proc. Int. Conf. Principles and Practice of Constraint Prog. (CP). pp. 137–151 (2010)
5. Chabert, G., Jaulin, L.: Contractor programming. Artif. Intell. **173**(11), 1079–1100 (2009)
6. Collavizza, H., Delobel, F., Rueher, M.: Extending consistent domains of numeric CSP. In: Proc. Int. Joint Conf. Artif. Intell. (IJCAI). pp. 406–413 (1999)
7. Fousse, L., Hanrot, G., Lefèvre, V., Pélissier, P., Zimmermann, P.: MPFR: a multiple-precision binary floating-point library with correct rounding. ACM Trans. Math. Softw. **33**(2) (2007)
8. Granvilliers, L.: A new interval contractor based on optimality conditions for bound constrained global optimization. In: Proc. Int. Conf. Tools with Artif. Intell. (ICTAI). pp. 90–97 (2018)
9. Granvilliers, L., Benhamou, F.: Algorithm 852: Realpaver: An interval solver using constraint satisfaction techniques. ACM Trans. Math. Softw. **32**(1), 138–156 (2006)
10. Hentenryck, P.V., McAllester, D., Kapur, D.: Solving polynomial systems using a branch and prune approach. SIAM J. Numer. Anal. **34**(2), 797–827 (1997)
11. Lhomme, O.: Consistency techniques for numeric CSPs. In: Proc. Int. Joint Conf. Artif. Intell. (IJCAI). pp. 232–238 (1993)
12. Mackworth, A.K.: Consistency in networks of relations. Artif. Intell. **8**, 99–118 (1977)
13. Moore, R.E.: Interval analysis. Prentice-Hall (1966)