A Box-Consistency Contraction Operator Based on extremal Functions

Gilles Trombettoni, Yves Papegay, Gilles Chabert, Odile Pourtallier

INRIA, Université de Nice-Sophia, France

October 2008, El Paso, Texas

Motivation

- New **interval-based** operators to better **solve non-linear systems of equations**.
- Two main types of algorithms: interval analysis (interval Newton), **constraint propagation**.
- Contribution: the `PolyBox` algorithm that:
  - improves the `Box-consistency` constraint propagation operator,
  - by taking advantage of the **symbolic form** of the handled equations.

Outline

1. Background: Contraction with constraint propagation
2. Box-consistency using extremal functions
3. Experiments

Constraint propagation

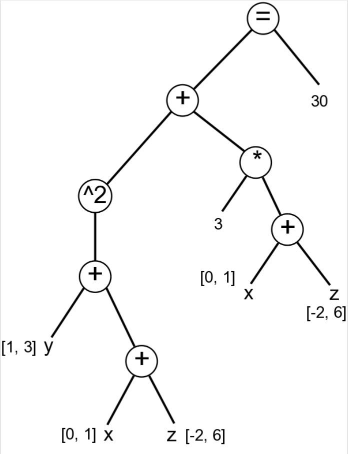
- **contraction** (filtering) algorithms from the constraint programming community used in interval solvers.
- Principle:
  - **Propagation**: At each step, a single equation is handled (by a so-called "revise" procedure), and the obtained reductions are propagated in the rest of the system until no interval can be reduced.
  - The **"revise"** procedure handles a single constraint. It reduces the intervals of the variables implied in the constraint.
- Two main constraint propagation algorithms: HC4 (Hull-consistency) and Box (Box-consistency).

HC4 Algorithm

- Every equation is represented by a binary tree. `HC4-Revise` works with this tree.
- `HC4-Revise` works in two phases: bottom-up **evaluation phase**, and top-down **narrowing phase**.
- `HC4-Revise` computes the 2B-consistency (or Hull-consistency) of the "decomposed" system of equations.
  Example: `HC4` applied to $(x + y + z)^2 + 3(x + z) = 30$ computes the 2B-consistency of the following decomposed system:
  $\{(x + y + z)^2 + 3(x_1 + z_1) = 30, x = x_1, z = z_1\}$

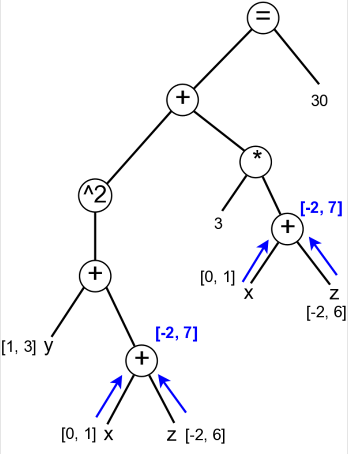
HC4-Revise algorithm

HC4-Revise applied to: $(x + y + z)^2 + 3(x + z) = 30$



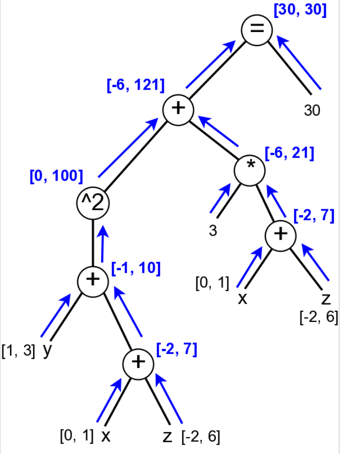
HC4-Revise algorithm

HC4-Revise applied to: $(x + y + z)^2 + 3(x + z) = 30$



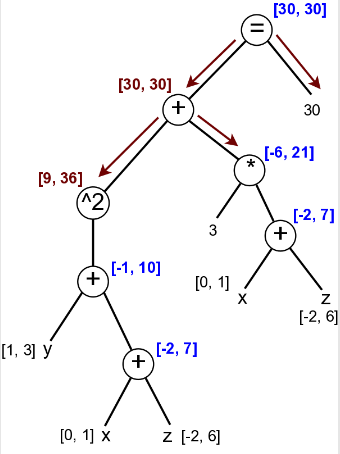
HC4-Revise algorithm

HC4-Revise applied to: $(x + y + z)^2 + 3(x + z) = 30$



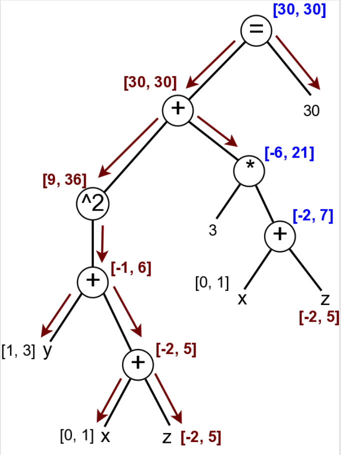
HC4-Revise algorithm

HC4-Revise applied to: $(x + y + z)^2 + 3(x + z) = 30$



HC4-Revise algorithm

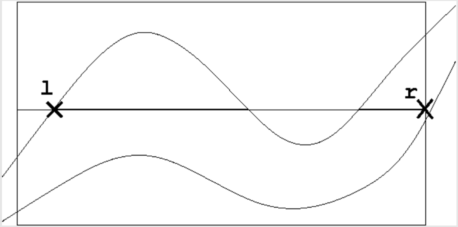
HC4-Revise applied to: $(x + y + z)^2 + 3(x + z) = 30$



BoxNarrow (or Box-Revise) Algorithm

- Box-Revise is the "revise" procedure of the Box algorithm.
- Box-Revise is applied to a pair (equation $f = 0$, variable x). A considered equation is $f(a, x) = 0$. $f : I\!R^{n-1} \times I\!R \to I\!R$ (a is a vectorial variable).
- Box-Revise works with the univariate interval (multivalued) function $f_{[a]}(x) = f([a], x)$: **variables in a are replaced by their current interval**.
- Starting from an initial interval $[x]$ for x, Box-Revise computes a contracted interval $[l, r]$ for x with no loss of solution.
- To do so, Box-Revise computes l (respectively, r) as the leftmost (respectively, the rightmost) root of $f_{[a]}(x) = 0$.

Example



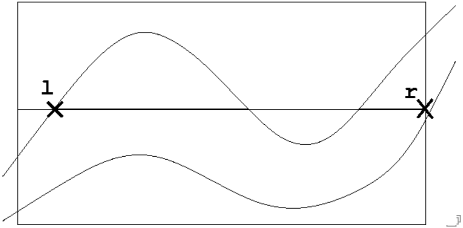
Two types of Revise procedures

- If x appears only once in the symbolic form of $f_{[a]}(x)$: `HC4-Revise` is optimal.
- If x appears several times in $f_{[a]}(x)$: To contract $[x]$, `Box-Revise` splits $[x]$ in slices $[x]_k$ and checks whether $[x]_k$ contains a root of $f_{[a]}(x) = 0$), with:
  - evaluations with the natural extension of $f_{[a]}$: $0 \in f_{[a]}([x]_k)$?
  - a univariate interval Newton.

Outline

Extremal functions

- The `Box-Revise` procedure may converge slowly because $f_{[a]}(x)$ is an interval (“thick”) function.
- Transform $f_{[a]}(x)$ into a new analytic form $g_{[a]}(x)$ such that **extremal functions** of $g_{[a]}(x)$ can be extracted quickly.
- Extremal functions:
  - Minimal function: $\underline{g_{[a]}}(x) = \min_{a_s \in [a]} f(a_s, x)$
  - Maximal function: $\overline{g_{[a]}}(x) = \max_{a_s \in [a]} f(a_s, x)$



Extraction of extremal functions

Favorable **expanded** form: $g_{[a]}(x) = \sum_{i=0}^{i=k} f_i([a]) \times h_i(x)$, with:

- $h_i(x)$ depends only on x, and not on a.
- $h_i(x)$ has a sign that does "not often" change: x^i, $log(x)$, e^x.

$\Rightarrow \underline{g_{[a]}}(x)$ and $\overline{g_{[a]}}(x)$ are piecewise functions with **punctual** coefficients in $\{\overline{f_i([a])}, \underline{f_i([a])}\}$.

First implementation:

$f_{[a]}(x)$ is a polynomial and $g_{[a]}(x) = \sum_{i=0}^{i=d} f_i([a]) \times x^i$ (d is the degree of the polynomial).

- i is even: x^i is positive.
- i is odd: the sign of x^i may change once (at $x = 0$).

Example

$$f(\{y,z\},x) = (y+z)\times x^2 + (2yz)\times x + sin(z)$$

$$g_{[a]}(x) = [-2,3]x^2 + [-4,-2]x + [-1,1]$$

- $x \geq 0$: $\overline{g_{[a]}}(x) = 3x^2 - 2x + 1$
- $x \leq 0$: $\overline{g_{[a]}}(x) = 3x^2 - 4x + 1$
- $x \geq 0$: $\underline{g_{[a]}}(x) = -2x^2 - 4x - 1$
- $x \leq 0$: $\underline{g_{[a]}}(x) = -2x^2 - 2x - 1$

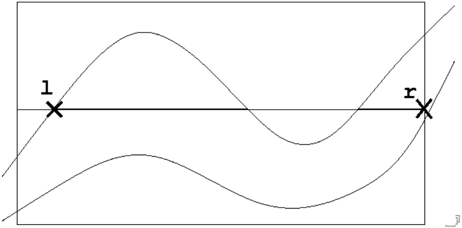
Key idea: Perform `Box-Revise` with the extremal punctual functions.

Contraction of $[x]$ with `PolyBoxRevise`

`PolyBoxRevise` determines a contracted interval $[l, r]$ of $[x]$. Let us detail the determination of l.

Step 1: With which extremal function to work?

- If $\underline{g}_{[a]}(\underline{[x]}) \leq 0$ and $0 \leq \overline{g}_{[a]}(\underline{[x]})$: $l = \underline{[x]}$ (no contraction)
- If $\underline{g}_{[a]}(\underline{[x]}) > 0$: we work with $\underline{g}_{[a]}$
- If $\overline{g}_{[a]}(\underline{[x]}) < 0$: we work with $\overline{g}_{[a]}$



Contraction of $[x]$ with `PolyBoxRevise`

Step 2: Computation of l

Two cases according to the degree d of $\overline{g_{[a]}}$:

- **High degree**: $d > 4$
  The smallest root of $\overline{g_{[a]}}(x) = 0$ inside $[x]$ is computed with `Box-Revise` applied to a **punctual** function.
- **Low degree**: $d \leq 4$
  The smallest root of $\overline{g_{[a]}}(x) = 0$ inside $[x]$ is extracted analytically.
  Implementation for degree 2 and degree 3 (with the method by Cardano).

Propagation with `PolyBoxRevise` (principles)

The “Revise” procedure selects the right routine according to $f_{[a]}(x)$:

- If $f_{[a]}(x)$ has no multiple occurrence of x: `HC4-Revise`
- Else `Mathematica` tries to transform $f_{[a]}(x)$ into $g_{[a]}(x) = \sum_{i=0}^{i=d} f_i([a]) \times x^i$.
  - If `Mathematica` fails, i.e., $f_{[a]}(x)$ is not polynomial: standard `Box-Revise` (or `HC4-Revise`)
  - Else If $g_{[a]}(x)$ has no multiple occurrence of x: `HC4-Revise` applied to $g_{[a]}(x)$.
    Example (`Caprasse`): $-2x + 2txy - z + y^2 z = 0 \Rightarrow x(-2 + 2ty) + (-1 + y^2)z = 0$
  - Else If the degree of $g_{[a]}(x)$ is low: the roots of one extremal function are determined analytically.
  - Else (the degree of $g_{[a]}(x)$ is high): the roots of one extremal function are determined numerically.

Symbolic form of $f_i(a)$

- Remark 1: A given function f can be rewritten in several forms according to the variable x contracted by the `PolyBoxRevise` procedure.
  Example: $f(t,x,y,z) = -2x + 2txy - z + y^2z = 0$
  $g(\{x,y,z\},t) = -2x + 2txy - z + y^2z = f(t,x,y,z) = 0$
  $g(\{t,y,z\},x) = (-2+2ty)x + (-1+y^2)z = 0$
  $g(\{t,x,z\},y) = -2x + 2txy - z + y^2z = f(t,x,y,z) = 0$
  $g(\{t,x,y\},z) = x(-2+2ty) + (-1+y^2)z = 0$
- Remark 2:
  The symbolic form of $f_i(a)$ obtained by Mathematica is crucial! The number of occurrences of the variables of a in $f_i(a)$ should be (generally) “minimized”.
- Good example: see above
- Bad example: Equation of `6body`:
  $5(B-D) + 3(b-d)(B+D-2F) = 0 \Rightarrow$
  $B(5+3(b-d)) + (-5+3b-3d)D + 6(-b+d)F = 0$

New contributions

P. Van Hentenryck et al. used the same idea in their interval-based solver `Numerica`. Main differences:

- `Numerica` used different systems of equations ("natural", Taylor form and expanded form). Propagation was performed separately on the system where equations has an expanded form. `PolyBox` manages a single system with different revise procedures.
- For a given equation, `PolyBox` may use a **different symbolic form** for every variable to be contracted. These (sophisticated) transformations are obtained with a **symbolic computation** tool.
- Contrarily to `Numerica`, `PolyBox` also uses `HC4-Revise` and **LowDegree**`PolyBoxRevise`.
- **Experimental comparison**: `PolyBox` vs `Box`.

Outline

Benchmarks

We compare `PolyBox` to `HC4` and `Box`.

Benchmarks

Among the 44 polynomial systems (with isolated solutions) found in the Web page of the COPRIN team, we have selected the 12 instances that:

- are solved by at least one of the 3 strategies in a time comprised between 1 second and 1 hour (`Pentium 3 GHz`),
- have equations with multiple occurrences of the variables.

Implementation

Solving strategy

- Variables are bisected with a round-robin strategy.
- Between two bisections:
  1. Constraint propagation (`PolyBox`, `HC4` or `Box` (`BC3`)).
  2. Interval Newton: Hansen-Sengupta, preconditioning of the matrix, Gauss-Seidel.

Implementation with `Mathematica` and with the free `Ibex/C++` interval-based solver (by Gilles Chabert).

Comparison between `PolyBox`, `HC4` and `Box`

Table: CPU time in second and number of generated boxes

Name	**#variables**	**#solutions**	`HC4`	`BC3`	`PolyBox`
`Caprasse`	4	18	5.53 9539	37.2 6509	**2.16** 2939
`Yamamura1`	8	7	34.3 42383	13.4 4041	**2.72** 2231
`Extended Wood`	4	3	**0.76** 4555	1.94 1947	1.12 3479
`Broyden Banded`	20	1	> 3600 ?	0.62 1	**0.09** 1
`Extended Freudenstein`	20	1	> 3600 ?	0.19 121	**0.11** 121
`6body`	6	5	**0.58** 4899	2.93 4797	0.73 4887
`Rose`	3	18	> 3600 ?	> 3600 ?	**4.11** 12521
`Discrete Boundary`	39	1	179 185617	29.5 3279	**16.1** 3281
`Katsura`	12	7	**102** 14007	404 11371	104 13719
`Eco9`	8	16	**66.4** 132873	191 125675	71 131911
`Broyden Tridiagonal`	20	2	470 269773	495 163787	**349.6** 164445
`Geneig`	6	10	3657 79472328	> 7200 ?	**3363** 4907705

Interest of low degree PolyBox

Table: CPU time in second. Left: PolyBox with **no** LowDegreepolyBoxRevise for polynomials of degrees 2 and 3. Right: PolyBox

Name	**#variables**	**#solutions**	PolyBox--	PolyBox
Caprasse	4	18	2.34	2.16
Yamamura1	8	7	5.79	**2.72**
Extended Wood	4	3	1.34	1.12
Broyden Banded	20	1	0.16	**0.09**
Extended Freudenstein	20	1	0.22	**0.11**
6body	6	5	0.72	0.73
Rose	3	18	3.95	4.11
Discrete Boundary	39	1	41.8	**16.1**
Katsura	12	7	103	104
Broyden Tridiagonal	20	2	403	**350**

Conclusion

- `PolyBox` is a step towards a unified local consistency algorithm.
- Proposition: a `PolyHC4` algorithm (no call to a costly standard `Box-Revise`)
- Future work: improve the symbolic form of the equations.
- Future work: extend the class of functions that are "tractable" with their extremal functions.

A Box-Consistency Contraction Operator Based on extremal Functions

Gilles Trombettoni, Yves Papegay, Gilles Chabert, Odile Pourtallier

INRIA, Université de Nice-Sophia, France

October 2008, El Paso, Texas