Interval Multivalued Inverse Functions

Relational Interval Arithmetic and its Use

Frédéric Goualard

`Frederic.Goualard@univ-nantes.fr`

Laboratoire d'Informatique de Nantes-Atlantique, UMR CNRS 6241

Historical motivation

Interval Newton operator:

$$N(\boldsymbol{d}) \doteq \boldsymbol{d} \cap \left(\mathsf{m}(\boldsymbol{d}) - \frac{\boldsymbol{f}(\mathsf{m}(\boldsymbol{d}))}{\boldsymbol{f'}(\boldsymbol{d})} \right)$$

Historical motivation

Interval Newton operator:

$$N(\boldsymbol{d}) \doteq \boldsymbol{d} \cap \left(\mathsf{m}(\boldsymbol{d}) - \frac{\boldsymbol{f}(\mathsf{m}(\boldsymbol{d}))}{\boldsymbol{f}'(\boldsymbol{d})} \right)$$

What if $0 \in \boldsymbol{f}'(\boldsymbol{d})$?

Historical motivation

Interval Newton operator:

$$N(\boldsymbol{d}) \doteq \boldsymbol{d} \cap \left(\mathsf{m}(\boldsymbol{d}) - \frac{\boldsymbol{f}(\mathsf{m}(\boldsymbol{d}))}{\boldsymbol{f}'(\boldsymbol{d})} \right)$$

What if $0 \in \boldsymbol{f}'(\boldsymbol{d})$?

[Moore, 1966] : $\boldsymbol{f}'(\boldsymbol{d})$ shall not contain 0

$$\boldsymbol{d_x}/\boldsymbol{d_y} = \{\gamma \in \mathbb{R} \mid \exists \alpha \in \boldsymbol{d_x} \exists \beta \in \boldsymbol{d_y} : \gamma = \alpha/\beta\}, \quad 0 \notin \boldsymbol{d_y}$$

[Alefeld, 1968, Hanson, 1968, Kahan, 1968] Extended interval arithmetic

[Walster, 1998] Closed Interval Arithmetic (containment sets)

Historical motivation

Interval Newton operator:

$$N(\boldsymbol{d}) \doteq \boldsymbol{d} \cap \left(\mathsf{m}(\boldsymbol{d}) - \frac{\boldsymbol{f}(\mathsf{m}(\boldsymbol{d}))}{\boldsymbol{f}'(\boldsymbol{d})} \right)$$

What if $0 \in \boldsymbol{f}'(\boldsymbol{d})$?

[Moore, 1966] : $\boldsymbol{f}'(\boldsymbol{d})$ shall not contain 0

$$\boldsymbol{d_x}/\boldsymbol{d_y} = \{\gamma \in \mathbb{R} \mid \exists \alpha \in \boldsymbol{d_x} \exists \beta \in \boldsymbol{d_y} \colon \gamma = \alpha/\beta\}, \quad 0 \notin \boldsymbol{d_y}$$

[Alefeld, 1968, Hanson, 1968, Kahan, 1968] Extended interval arithmetic

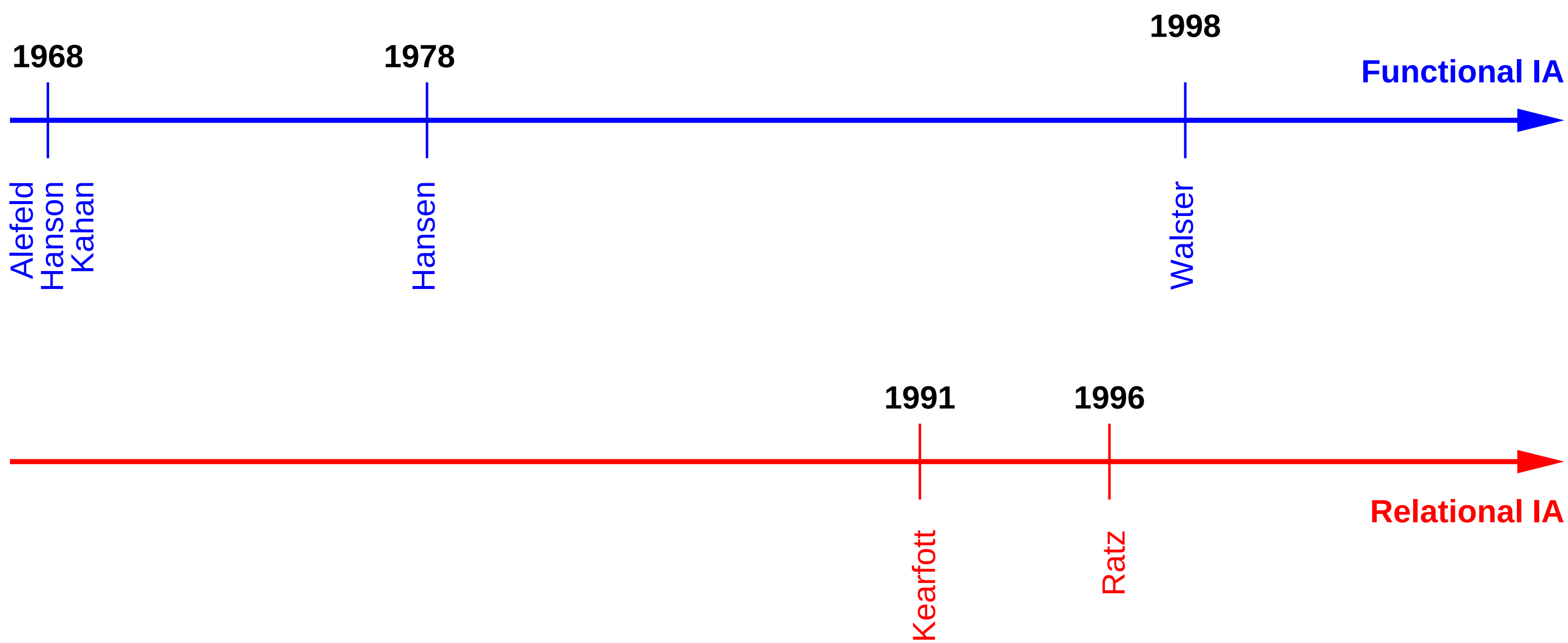
[Walster, 1998] Closed Interval Arithmetic (containment sets)

[Ratz, 1996] Extended interval division ($z = x/y$ considered as a ***relation***)

$$\boldsymbol{d_x} \oslash \boldsymbol{d_y} = \mathrm{cch}(\{\gamma \in \mathbb{R} \mid \exists \alpha \in \boldsymbol{d_x} \exists \beta \in \boldsymbol{d_y} \colon \beta\gamma = \alpha\})$$

Return to “original” problem: $\boldsymbol{f}(\mathsf{m}(\boldsymbol{d})) + \boldsymbol{f}'(\boldsymbol{d})(x - \mathsf{m}(\boldsymbol{d})) = 0$

Timeline



- [Ratz, 1996]: “relational” division (credits Kearfott)
- [Kearfott, 1991]: multivalued inverse of elementary functions ((NL)GS)

Relational division and Gauss-Seidel

Given the linear constraint:

$$\boldsymbol{a_1} x_1 + \cdots + \boldsymbol{a_n} x_n = \boldsymbol{b} \tag{1}$$

Inner step of Gauss-Seidel (symbolic inversion on x_i):

$$x_i = \frac{\boldsymbol{b} - \boldsymbol{a_1} x_1 + \cdots + \boldsymbol{a_{i-1}} x_{i-1} + \boldsymbol{a_{i+1}} x_{i+1} + \cdots + \boldsymbol{a_n} x_n}{\boldsymbol{a_i}} \tag{2}$$

Eq. (2) and Eq. (1) are “equivalent” when using relational division

The next step

What if we had relational versions of inverse power, inverse cosine, . . . ?

The next step

What if we had relational versions of inverse power, inverse cosine, . . . ?

$$3x^4 - 5xy^2 + 6z = 1 \quad \leadsto \quad y = \sqrt[\text{rel}]{\frac{3x^4+6z-1}{5x}}$$

$$\cos x + 3z \tan y = 2 \quad \leadsto \quad x = \cos^{-1}(2 - 3z \tan y)$$

$$\vdots$$

$$f(x_1, \ldots, x_n) = 0 \quad \leadsto \quad x_i = g(x_1, \ldots, x_{i-1}, x_{i+1}, x_n)$$

Symbolic inversion of nonlinear constraints

The next step

What if we had relational versions of inverse power, inverse cosine, . . . ?

$$
\begin{aligned}
3x^4 - 5xy^2 + 6z = 1 \quad &\rightsquigarrow \quad y = \sqrt[\text{rel}]{\tfrac{3x^4+6z-1}{5x}} \\
\cos x + 3z \tan y = 2 \quad &\rightsquigarrow \quad x = \cos^{-1}(2 - 3z \tan y) \\
&\vdots \\
f(x_1, \ldots, x_n) = 0 \quad &\rightsquigarrow \quad x_i = g(x_1, \ldots, x_{i-1}, x_{i+1}, x_n)
\end{aligned}
$$

Symbolic inversion of nonlinear constraints

$\rightsquigarrow$ nonlinear Gauss-Seidel step

Nonlinear Gauss-Seidel

$\mathsf{GS}\big($**in** $F = (f_1, \ldots, f_n) \colon \mathbb{R}^n \to \mathbb{R}^n$; **inout** $\boldsymbol{B} = \boldsymbol{d_1} \times \cdots \times \boldsymbol{d_n} \in \mathbb{I}^n\big)$
begin
    modified $\leftarrow$ **true**;
    $\boldsymbol{B'} \leftarrow [-\infty, +\infty]^n$
    while $\mathsf{w}(\boldsymbol{B}) > \varepsilon$ **and modified do**
        $\boldsymbol{B'} \leftarrow \boldsymbol{B}$
        for $j = 1$ **to** n **do**
            $\boldsymbol{d_j} \leftarrow \boldsymbol{d_j} \cap \mathsf{tighten}(f_j, v_j, \boldsymbol{B})$
        endfor
        modified $\leftarrow (\mathsf{dist}(\boldsymbol{B}, \boldsymbol{B'}) > \Delta)$
    endwhile
end

Nonlinear Gauss-Seidel

$\mathsf{GS}\big($**in** $F = (f_1, \ldots, f_n)\colon \mathbb{R}^n \to \mathbb{R}^n$; **inout** $\boldsymbol{B} = \boldsymbol{d_1} \times \cdots \times \boldsymbol{d_n} \in \mathbb{I}^n\big)$

begin

  modified $\leftarrow$ **true**;

  $\boldsymbol{B'} \leftarrow [-\infty, +\infty]^n$

  while $\mathsf{w}(\boldsymbol{B}) > \varepsilon$ **and modified do**

    $\boldsymbol{B'} \leftarrow \boldsymbol{B}$

    for $j = 1$ **to** n **do**

      $\boldsymbol{d_j} \leftarrow \boldsymbol{d_j} \cap \mathsf{tighten}(f_j, v_j, \boldsymbol{B})$

    endfor

    modified $\leftarrow (\mathsf{dist}(\boldsymbol{B}, \boldsymbol{B'}) > \Delta)$

  endwhile

end

- $f_j(x_1, \ldots, x_n) = a_1 x_1 + \cdots + a_n x_n$
  $\rightsquigarrow \mathsf{tighten}(f_j, x_j, \boldsymbol{B})\colon \boldsymbol{d_j} \leftarrow \boldsymbol{d_j} \cap \frac{\boldsymbol{d_b} - \boldsymbol{a_1 d_1} + \cdots + \boldsymbol{a_{j-1} d_{j-1}} + \boldsymbol{a_{j+1} d_{j+1}} + \cdots + \boldsymbol{a_n d_n}}{\boldsymbol{a_j}}$

Nonlinear Gauss-Seidel

GS(**in** $F = (f_1, \ldots, f_n) \colon \mathbb{R}^n \to \mathbb{R}^n$; **inout** $\boldsymbol{B} = \boldsymbol{d_1} \times \cdots \times \boldsymbol{d_n} \in \mathbb{I}^n$)
begin
  modified $\leftarrow$ **true**;
  $\boldsymbol{B'} \leftarrow [-\infty, +\infty]^n$
  while w$(\boldsymbol{B}) > \varepsilon$ **and modified do**
    $\boldsymbol{B'} \leftarrow \boldsymbol{B}$
    for $j = 1$ **to** n **do**
      $\boldsymbol{d_j} \leftarrow \boldsymbol{d_j} \cap$ tighten$(f_j, v_j, \boldsymbol{B})$
    endfor
    modified $\leftarrow$ (dist$(\boldsymbol{B}, \boldsymbol{B'}) > \Delta$)
  endwhile
end

- $f_j(x_1, \ldots, x_n)$ nonlinear
  $\rightsquigarrow$ tighten$(f_j, x_j, \boldsymbol{B})$: $\boldsymbol{d_j} \leftarrow \boldsymbol{d_j} \cap$ Newton$(f_j, x_j, \boldsymbol{B})$

Nonlinear Gauss-Seidel

$\mathsf{GS}\big($**in** $F = (f_1, \ldots, f_n)\colon \mathbb{R}^n \to \mathbb{R}^n$; **inout** $\boldsymbol{B} = \boldsymbol{d_1} \times \cdots \times \boldsymbol{d_n} \in \mathbb{I}^n\big)$
begin
  modified $\leftarrow$ **true**;
  $\boldsymbol{B'} \leftarrow [-\infty, +\infty]^n$
  while $\mathsf{w}(\boldsymbol{B}) > \varepsilon$ **and modified do**
    $\boldsymbol{B'} \leftarrow \boldsymbol{B}$
    for $j = 1$ **to** n **do**
      $\boldsymbol{d_j} \leftarrow \boldsymbol{d_j} \cap \mathsf{tighten}(f_j, v_j, \boldsymbol{B})$
    endfor
    modified $\leftarrow (\mathsf{dist}(\boldsymbol{B}, \boldsymbol{B'}) > \Delta)$
  endwhile
end

- $f_j(x_1, \ldots, x_n)$ nonlinear
  $\rightsquigarrow \mathsf{tighten}(f_j, x_j, \boldsymbol{B})\colon \boldsymbol{d_j} \leftarrow \boldsymbol{d_j} \cap [\mathsf{Invert}(f_j, x_j)](\boldsymbol{d_1} \times \cdots \times \boldsymbol{d_{j-1}} \times \boldsymbol{d_{j+1}} \times \cdots \times \boldsymbol{d_n})$

Explicit inversion

Solving $c\colon f(x_1, x_2) = x_1^3 + x_1^2 x_2 + x_2^2 + 1 = 0$ for x_1 or x_2:

- [Kearfott, 1991] Decompose c into easily invertible constraints:

$$x_1^3 + x_1^2 x_2 + x_2^2 + 1 = 0 \rightsquigarrow \left\{ \begin{array}{ll|ll} v_1 - x_1^3 & = 0 & v_5 - x_2^2 & = 0 \\ v_2 - x_1^2 & = 0 & v_6 - (v_4 + v_5) & = 0 \\ v_3 - v_2 x_2 & = 0 & v_7 - (v_6 + 1) & = 0 \\ v_4 - (v_1 + v_3) & = 0 & & \end{array} \right.$$

- [Ceberio and Granvilliers, 2000]: Recursively invert constraint expression with rules: $f[x] + g = h \rightsquigarrow f[x] = h - g$

$$\vdots$$

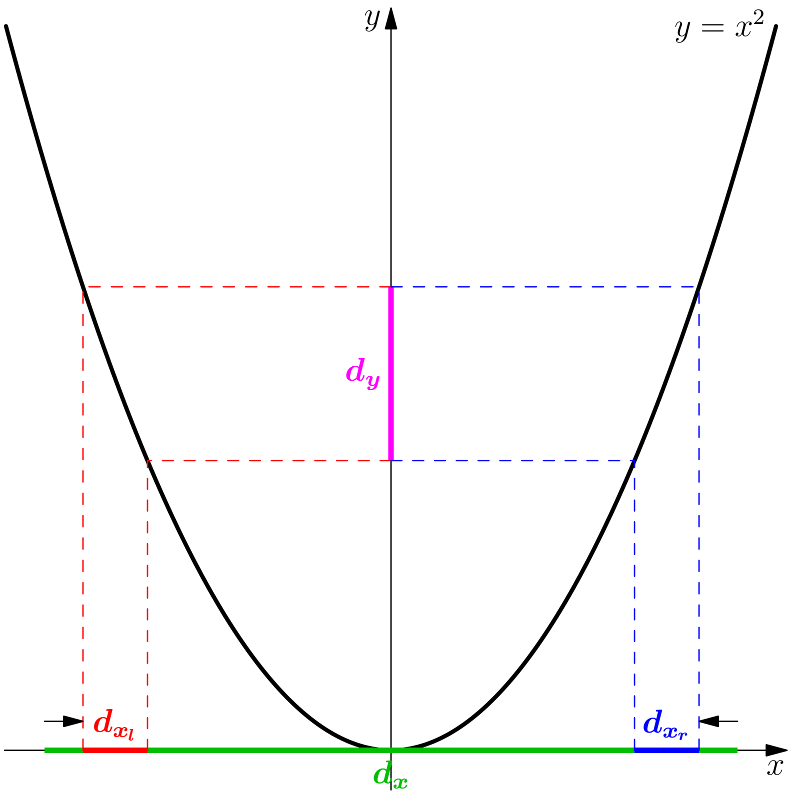
$$exp(f[x]) = h \rightsquigarrow f[x] = \log h$$

- [Hansen and Walster, 2003]: Express f as $g - h$, with g easily invertible for x_1 or x_2 and evaluate $g^{-1}(h)$, e.g.:

$$f(x_1, x_2) = x_1^2 x_2 - (-1 - x_2^2 - x_1^3) \rightsquigarrow \left\{ \begin{array}{ll} x_1 = & \sqrt{\frac{-1-x_2^2-x_1^3}{x_2}} \\ x_2 = & \frac{-1-x_2^2-x_1^3}{x_1^2} \end{array} \right.$$

Multivalued inverse function (1)

- Definition of new relational operators



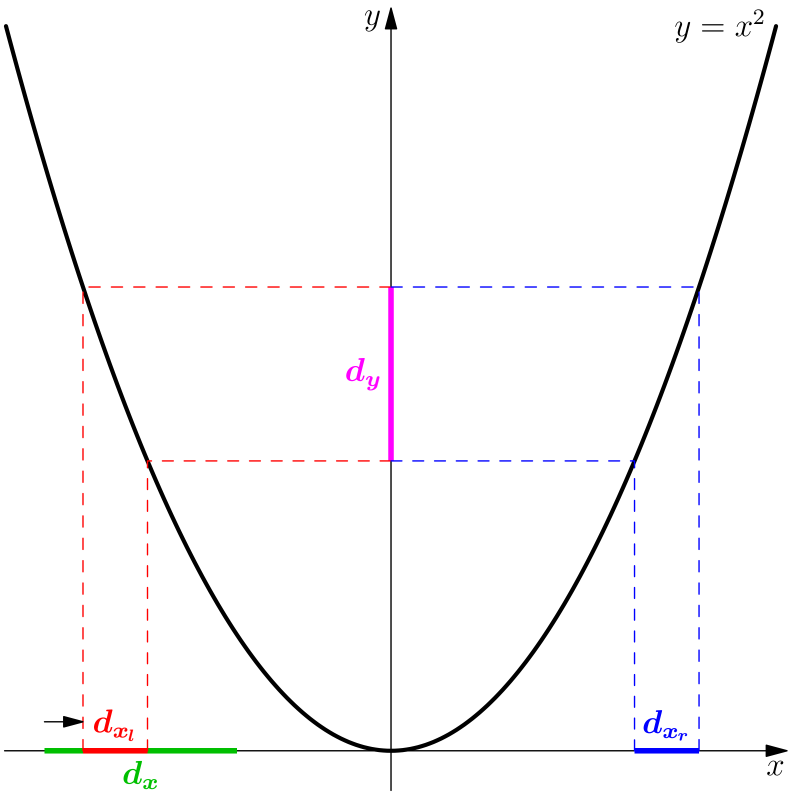
$$\begin{cases} x \in [-4.5, 4.5] \\ y \in [10, 16] \\ x^2 = y \end{cases}$$

$$\begin{cases} \boldsymbol{d_x} \leftarrow \boldsymbol{d_x} \cap \texttt{sqrt\_rel}(\boldsymbol{d_y}) \\ \boldsymbol{d_y} \leftarrow \boldsymbol{d_y} \cap \texttt{pow}(\boldsymbol{d_x}, 2) \end{cases}$$

$$\rightsquigarrow \begin{cases} x \in [-\sqrt{16}, \sqrt{16}] \\ y \in [10, 16] \end{cases}$$

Multivalued inverse function (1)

- Definition of new relational operators



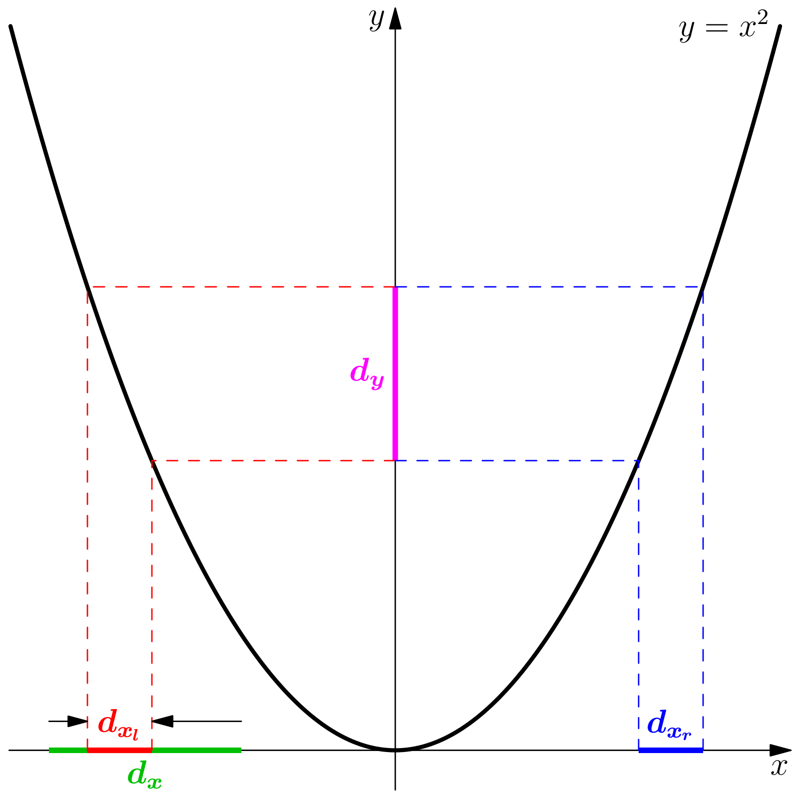
$$\begin{cases} x \in [-4.5, -2] \\ y \in [10, 16] \\ x^2 = y \end{cases}$$

$$\begin{cases} \boldsymbol{d_x} \leftarrow \boldsymbol{d_x} \cap \texttt{sqrt\_rel}(\boldsymbol{d_y}) \\ \boldsymbol{d_y} \leftarrow \boldsymbol{d_y} \cap \texttt{pow}(d_x, 2) \end{cases}$$

$$\rightsquigarrow \begin{cases} x \in [-\sqrt{16}, -2] \\ y \in [10, 16] \end{cases}$$

Multivalued inverse function (1)

- Definition of new relational operators



$$\begin{cases} x \in [-4.5, -2] \\ y \in [10, 16] \\ x^2 = y \end{cases}$$

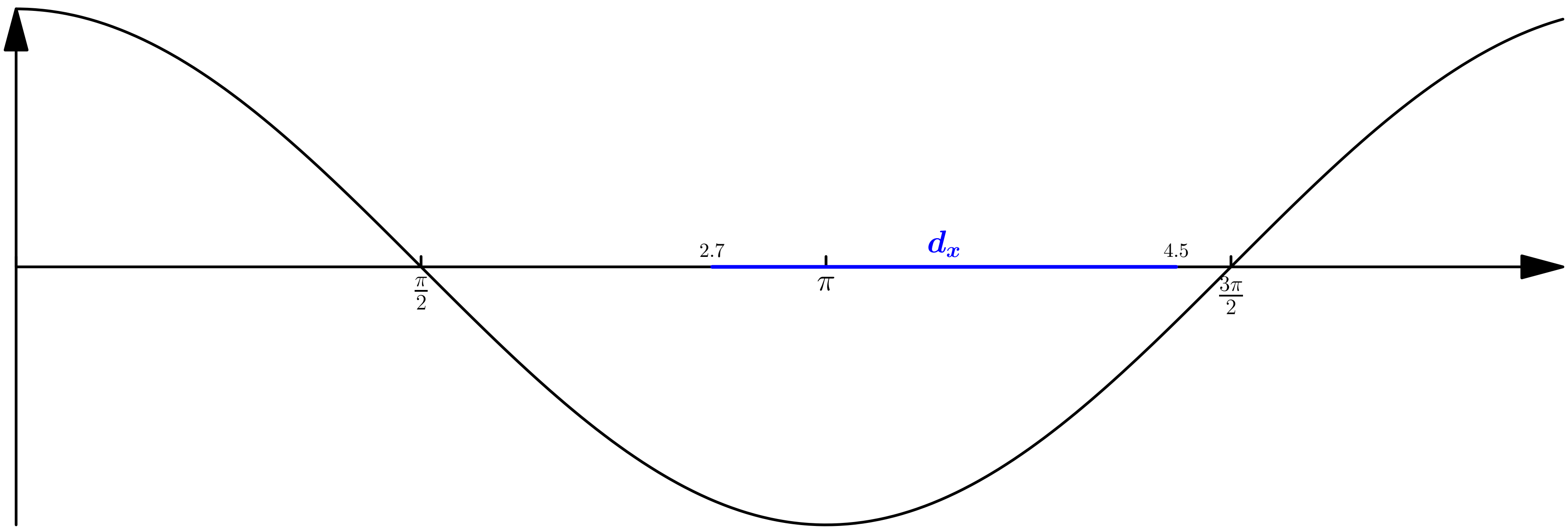
$$\begin{cases} \boldsymbol{d_x} \leftarrow \boldsymbol{d_x} \cap \texttt{sqrt\_rel}(\boldsymbol{d_y}, \boldsymbol{d_x}) \\ \boldsymbol{d_y} \leftarrow \boldsymbol{d_y} \cap \texttt{pow}(d_x, 2) \end{cases}$$

$$\rightsquigarrow \begin{cases} x \in [-\sqrt{16}, -\sqrt{10}] \\ y \in [10, 16] \end{cases}$$

Multivalued inverse function (2)

$$y = \cos x, \quad x \in \boldsymbol{d_x}$$

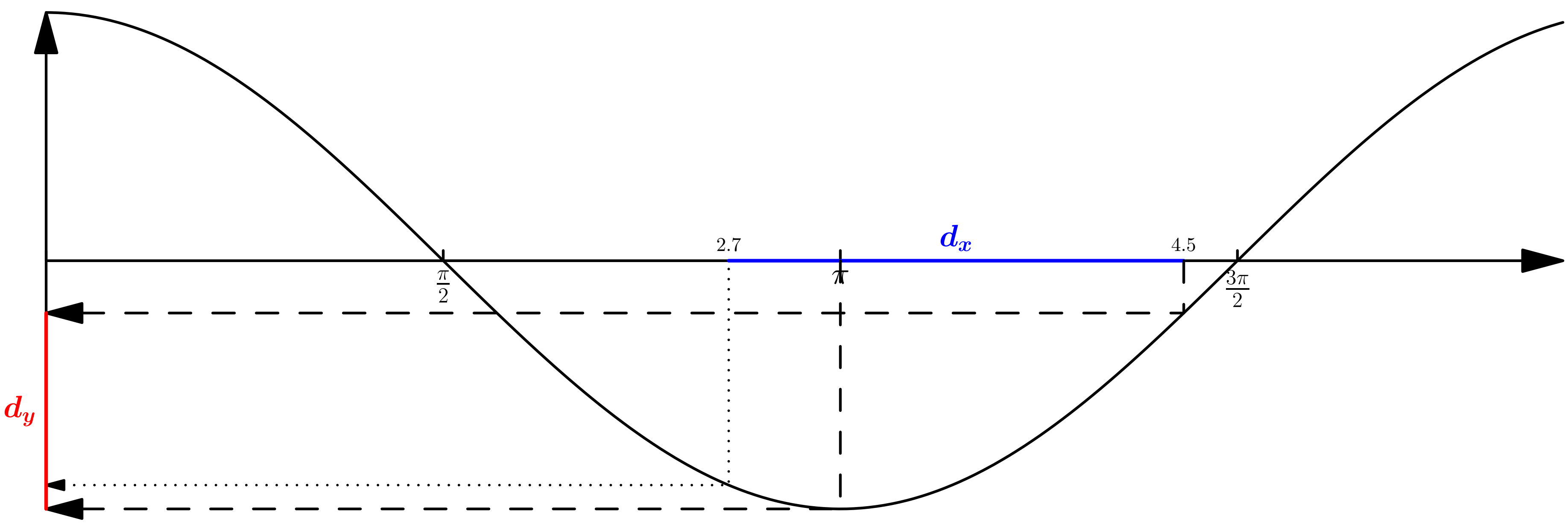
$\boldsymbol{d_y}$?



Multivalued inverse function (2)

$$y = \cos x, \quad x \in \boldsymbol{d_x}$$

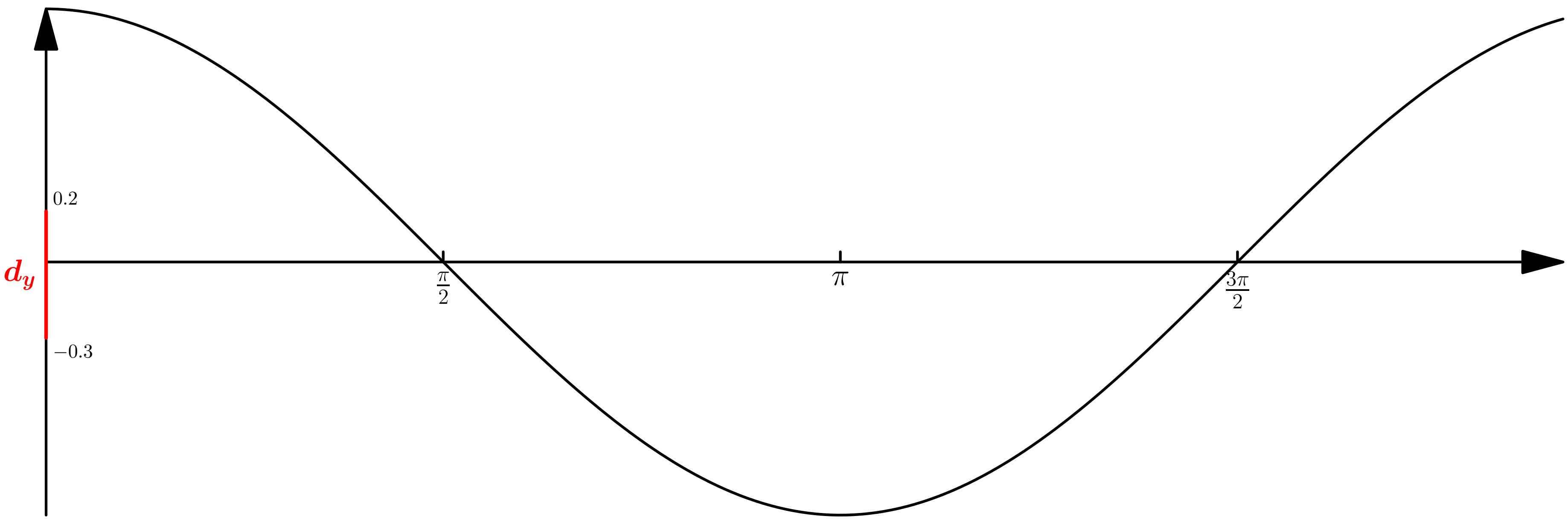
$$\boldsymbol{d_y} = \cos \boldsymbol{d_x} = [-1, \cos 4.5] = \text{cch}(\{\cos \alpha \mid \alpha \in \boldsymbol{d_x}\})$$



Multivalued inverse function (2)

$$y = \cos x, \quad y \in \boldsymbol{d_y}$$

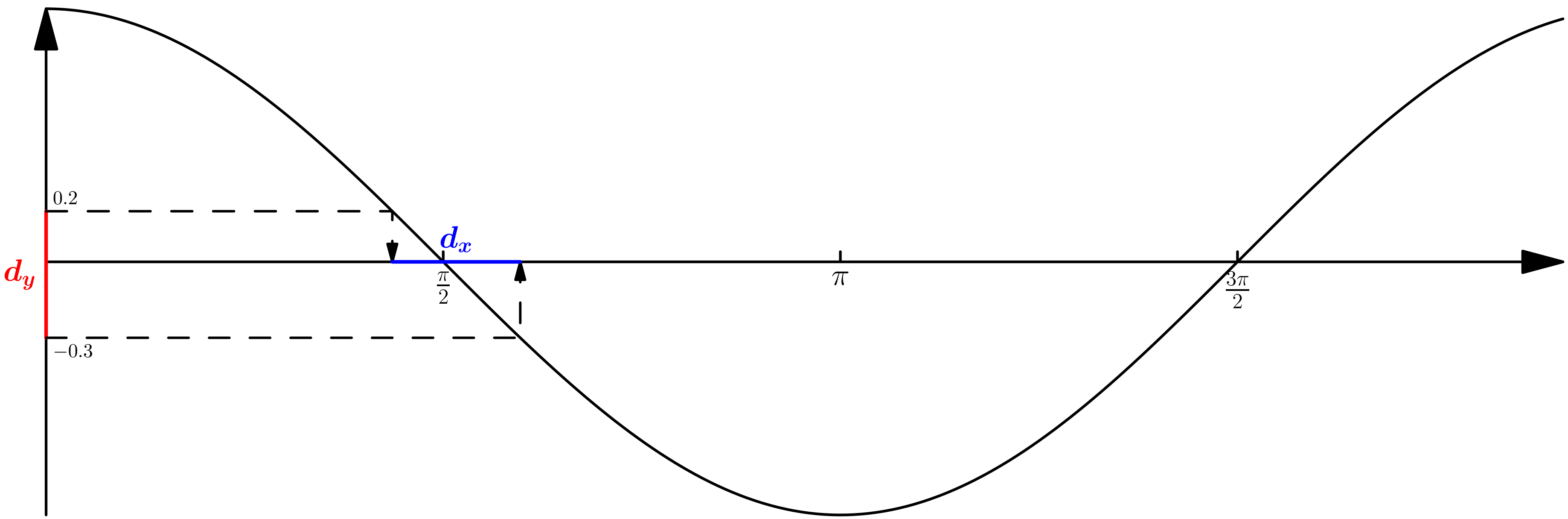
$\boldsymbol{d_x}$?



Multivalued inverse function (2)

$$y = \cos x, \quad y \in \boldsymbol{d_y}$$

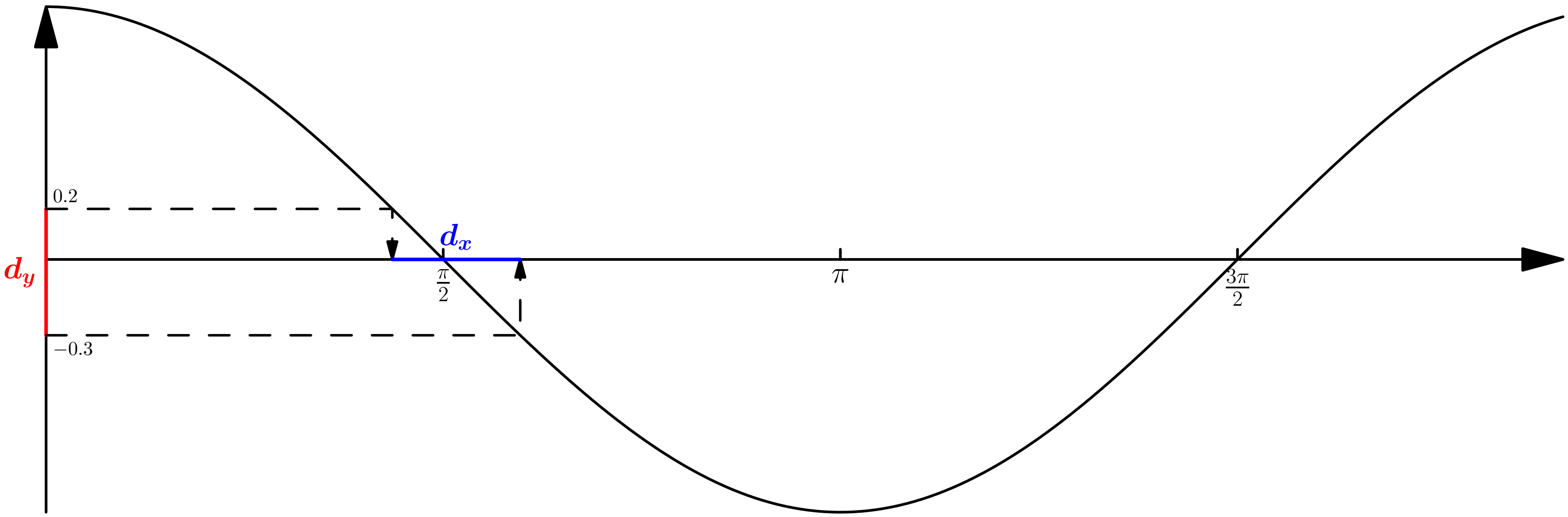
$$\boldsymbol{d_x} = \operatorname{acos} \boldsymbol{d_y} = [\operatorname{acos} 0.2, \operatorname{acos} -0.3] = \operatorname{cch}(\{\operatorname{acos} \beta \mid \beta \in \boldsymbol{d_y}\})$$



Multivalued inverse function (2)

$$y = \cos x, \quad y \in \boldsymbol{d_y}$$

What if $x \in [20, 24]$?

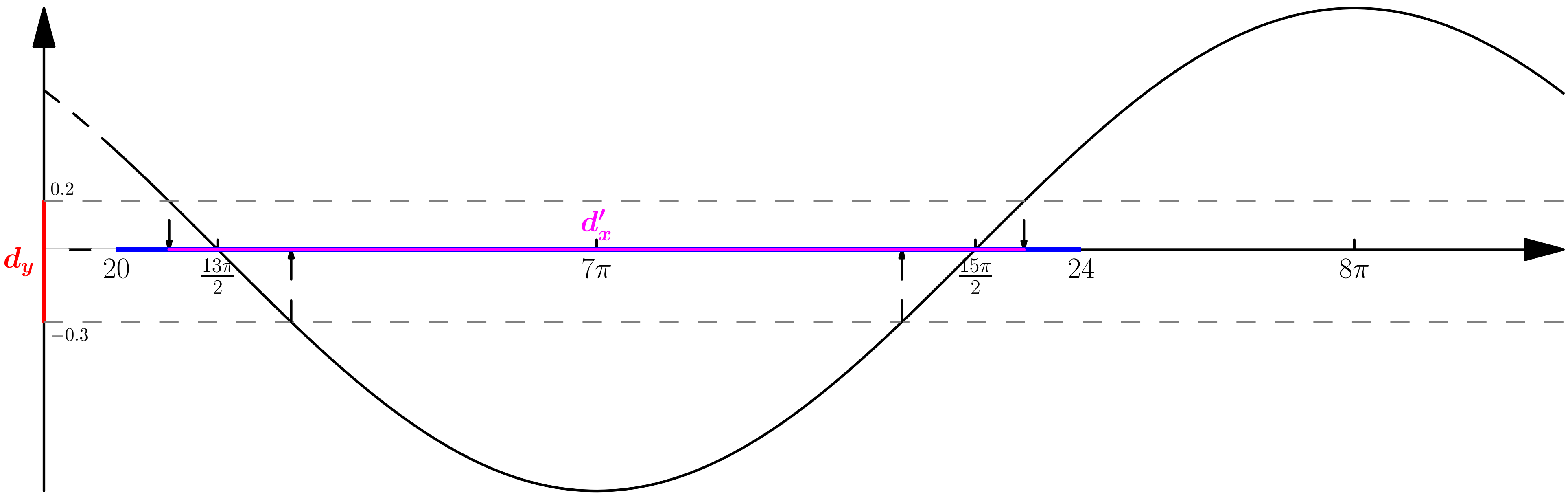


Multivalued inverse function (2)

$$y = \cos x, \quad y \in \boldsymbol{d_y}$$

What if $x \in [20, 24]$?

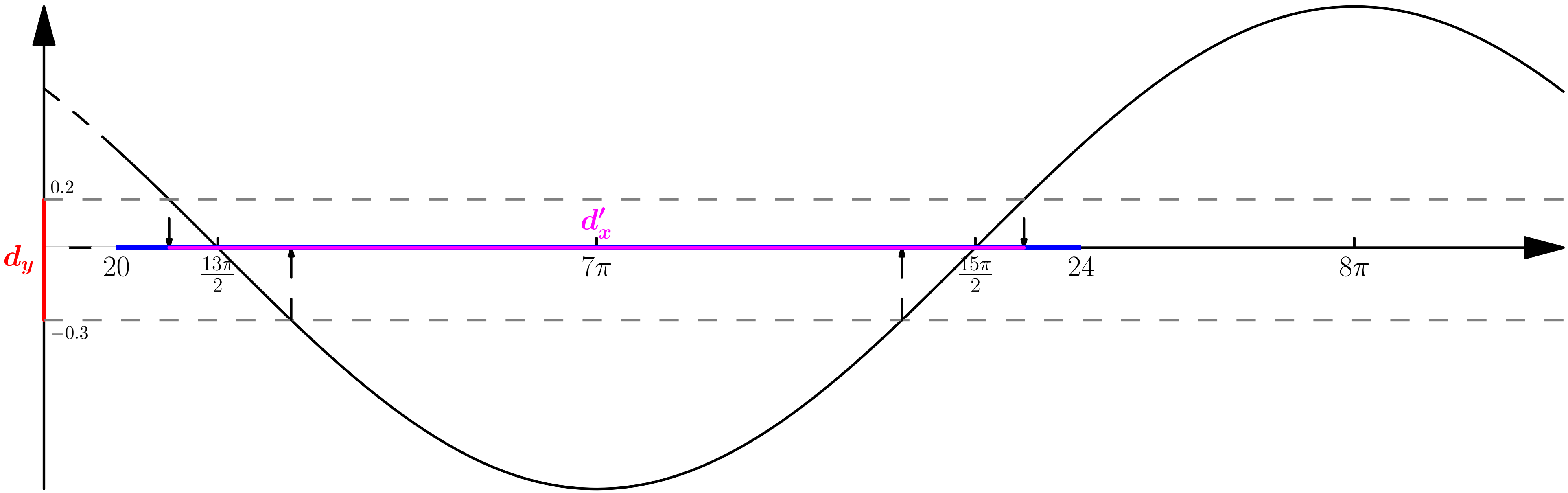
$$\boldsymbol{d'_x} = \text{acos\_rel}(\boldsymbol{d_y}, \boldsymbol{d_x}) = [6\pi + \operatorname{acos} 0.2, 8\pi - \operatorname{acos} 0.2]$$
$$)$$
$$= \operatorname{cch}(\{\alpha \in \boldsymbol{d_x} \mid \exists \beta \in \boldsymbol{d_y} : \beta = \cos \alpha\}$$



Multivalued inverse function (2)

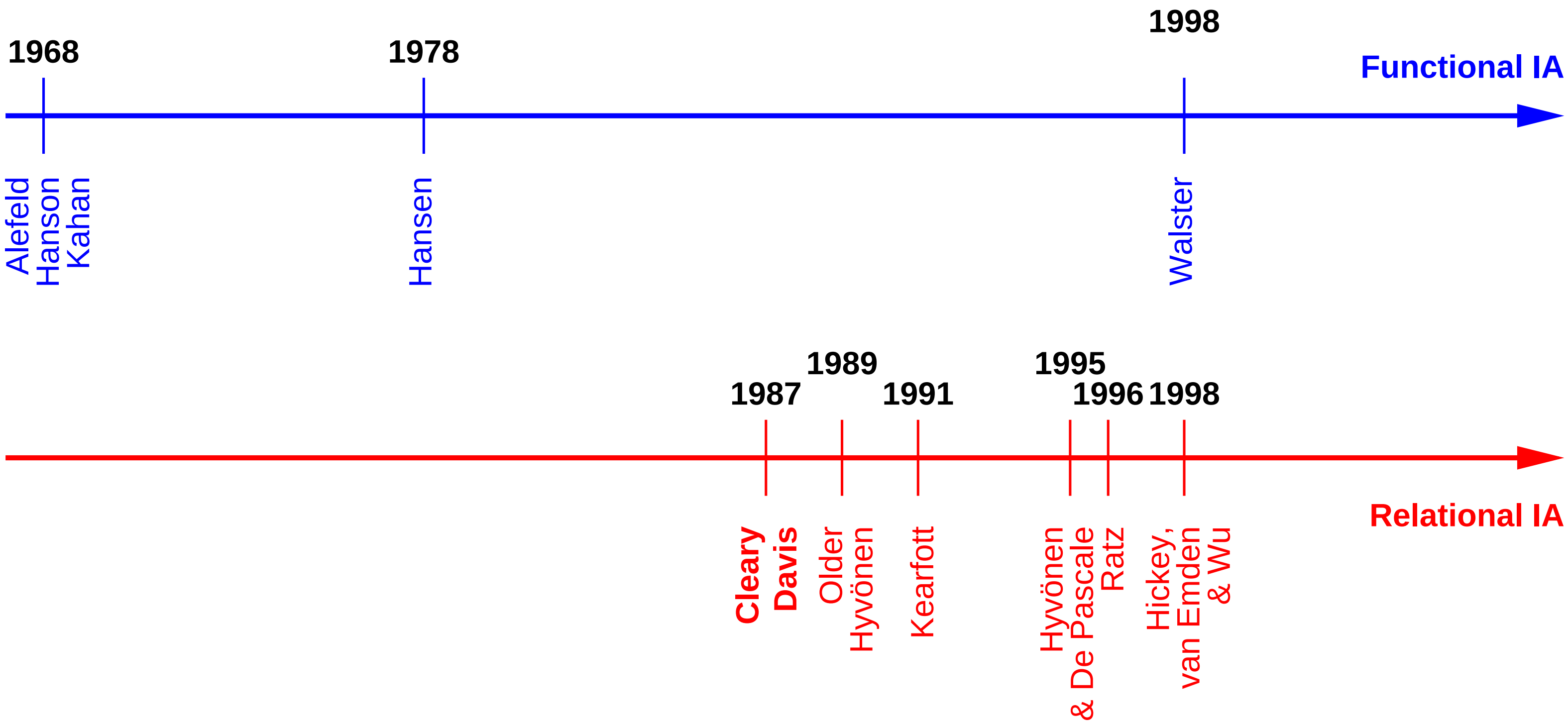
$$y = \cos x, \quad y \in \boldsymbol{d_y}$$

$$\begin{aligned} \boldsymbol{d'_x} &= \text{acos\_rel}(\boldsymbol{d_y}, \boldsymbol{d_x}) = [6\pi + \operatorname{acos} 0.2, 8\pi - \operatorname{acos} 0.2] \;) \\ &= \operatorname{cch}(\{\alpha \in \boldsymbol{d_x} \mid \exists \beta \in \boldsymbol{d_y} : \beta = \cos\alpha\}\end{aligned}$$



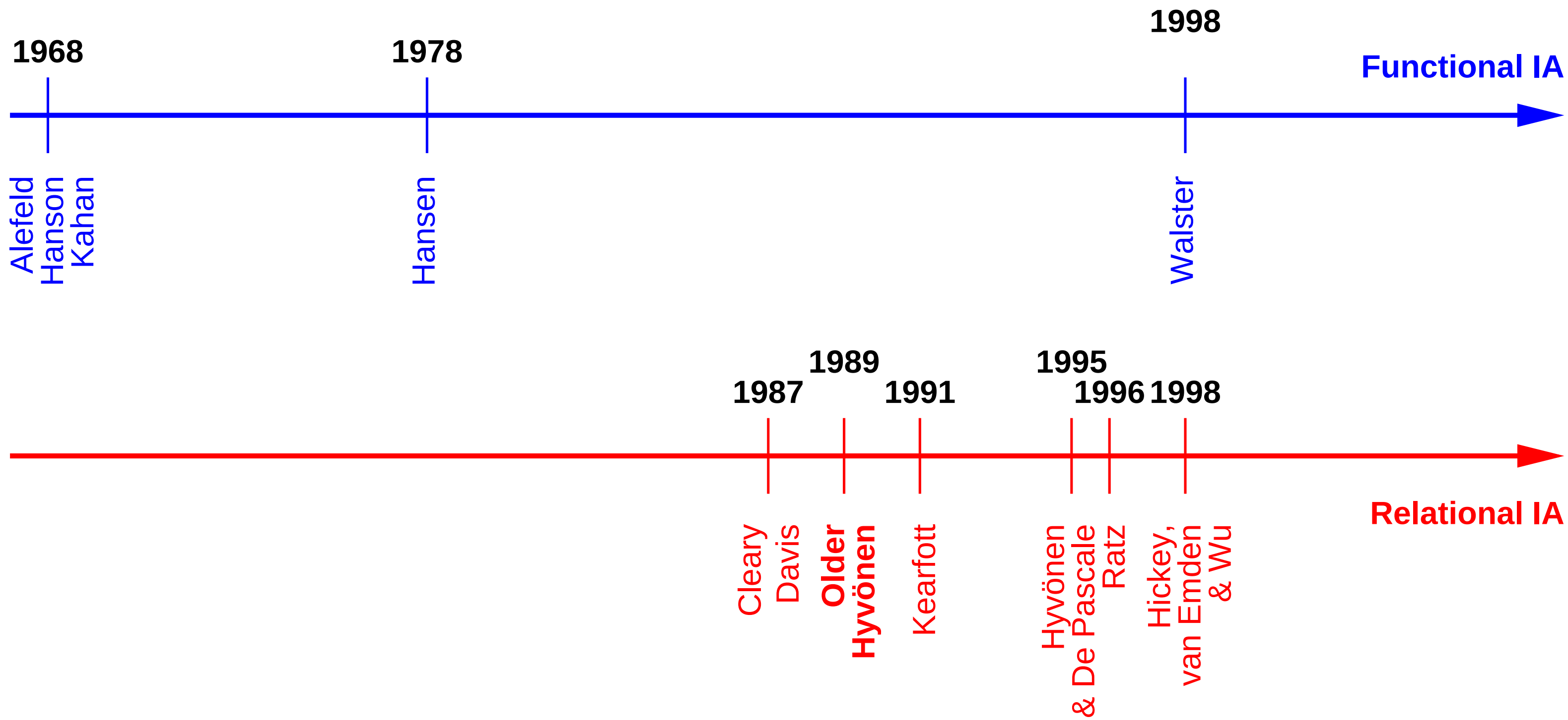
- To ensure tight results, define $(n+1)$-ary operators
  $\rightsquigarrow$ *Relational Arithmetic*

Timeline



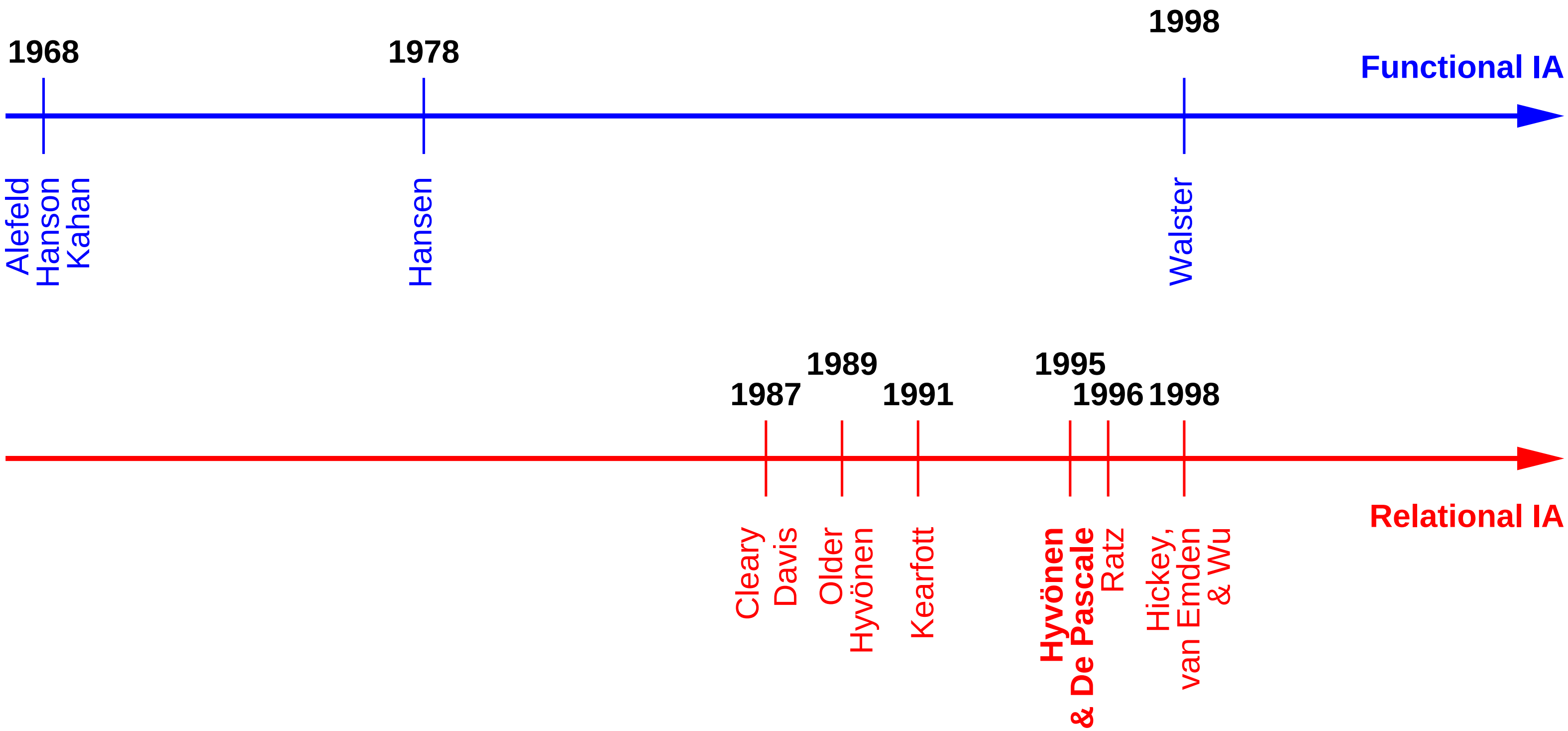
- [Cleary, 1987] : relational arithmetic

Timeline



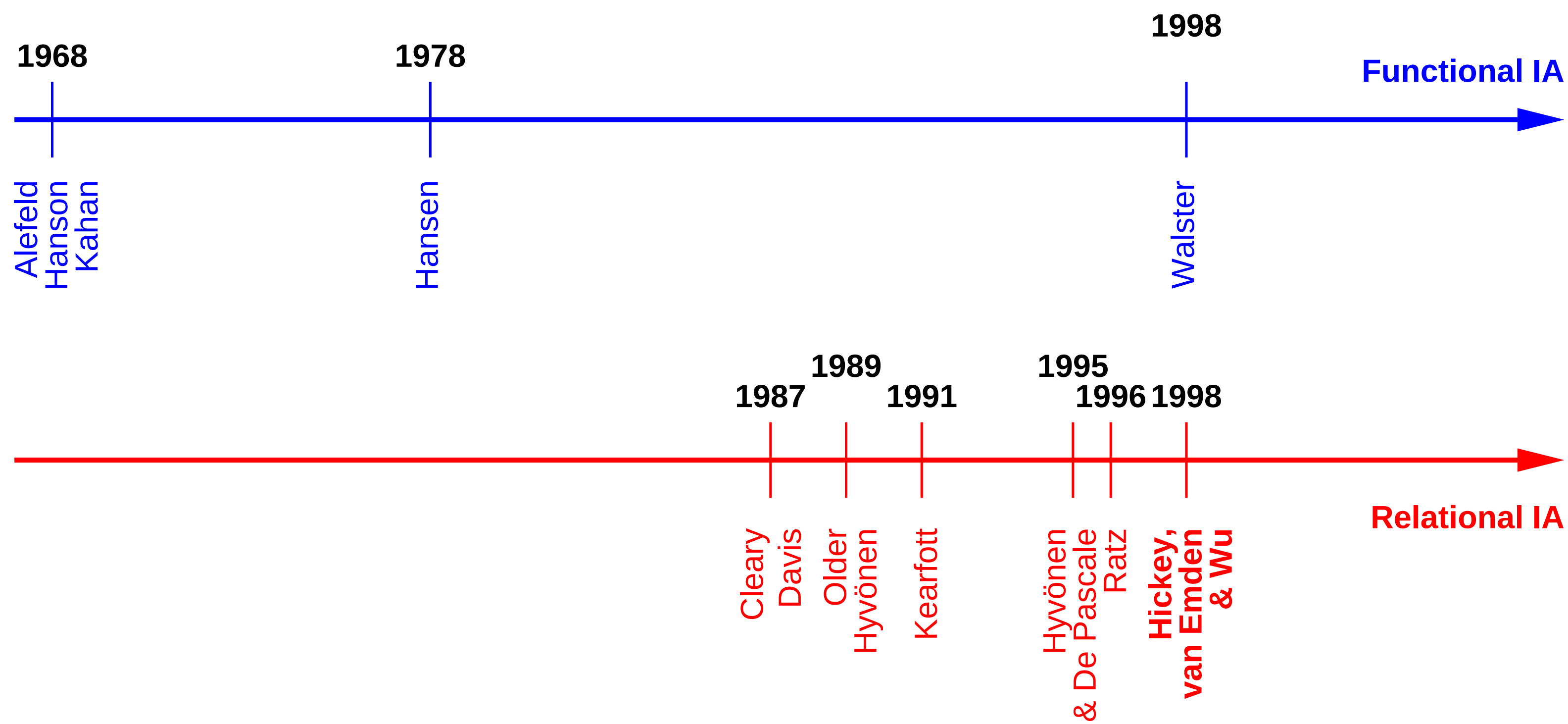
- [Cleary, 1987] : relational arithmetic
- [Older, 1989] : Interval constraint solver (BNR Prolog)

Timeline



- [Cleary, 1987] : relational arithmetic
- [Older, 1989] : Interval constraint solver (BNR Prolog)
- [Hyvönen and de Pascale, 1995] : (C++) library with relational operators

Timeline



- [Cleary, 1987] : relational arithmetic
- [Older, 1989] : Interval constraint solver (BNR Prolog)
- [Hyvönen and de Pascale, 1995] : (C++) library with relational operators
- [Hickey et al., 1998] : Formal description of relational division in context of relational arithmetic

Logical Arithmetic [Cleary, 1987] (1)

Prolog = programming with relations:

```
liege(arthur,caradoc).
liege(arthur,galahad).
:- liege(arthur,X).
:- liege(Y,galahad).
```

```
equation(X,Y):- Y is X*(X-2)+1.
:- equation(3,Y). # Y bound to 4
:- equation(X,0). # Failure!
```

Predicate "`is/2`" is essentially functional

Cleary disatisfied by actual implementation of arithmetic in Prolog

Logical arithmetic

- Use of intervals to enclose the value of real variables
- *All* operators become relations:

  ```
  equation(X,Y):- add(V1,2,X), mul(X,V1,V2), add(V2,1,Y).
  ```

Implementation (`mul/3`):

$$\begin{cases} \boldsymbol{d}_{\mathsf{X}} \leftarrow \boldsymbol{d}_{\mathsf{X}} \cap (\boldsymbol{d}_{\mathsf{Z}} \oslash \boldsymbol{d}_{\mathsf{Y}}) \\ \boldsymbol{d}_{\mathsf{d}} \leftarrow \boldsymbol{d}_{\mathsf{d}} \cap (\boldsymbol{d}_{\mathsf{Z}} \oslash \boldsymbol{d}_{\mathsf{X}}) \\ \boldsymbol{d}_{\mathsf{Z}} \leftarrow \boldsymbol{d}_{\mathsf{Z}} \cap (\boldsymbol{d}_{\mathsf{X}} \times \boldsymbol{d}_{\mathsf{Y}}) \end{cases}$$

Logical Arithmetic [Cleary, 1987] (2)

- Use of infinite bounds
- Intervals may be open-ended
- Unions of intervals (e.g., when dividing by an interval containing 0) handled through backtracking (each interval considered alternatively):

  `X=[-2,3], Y=[`$-\infty, +\infty$`], Z=[1,1], mul(X,Y,Z)`

  Y bound to `[-`∞`,-1/2]`, and then to `[1/3,+`∞`]`

BNR Prolog [Older, 1989]

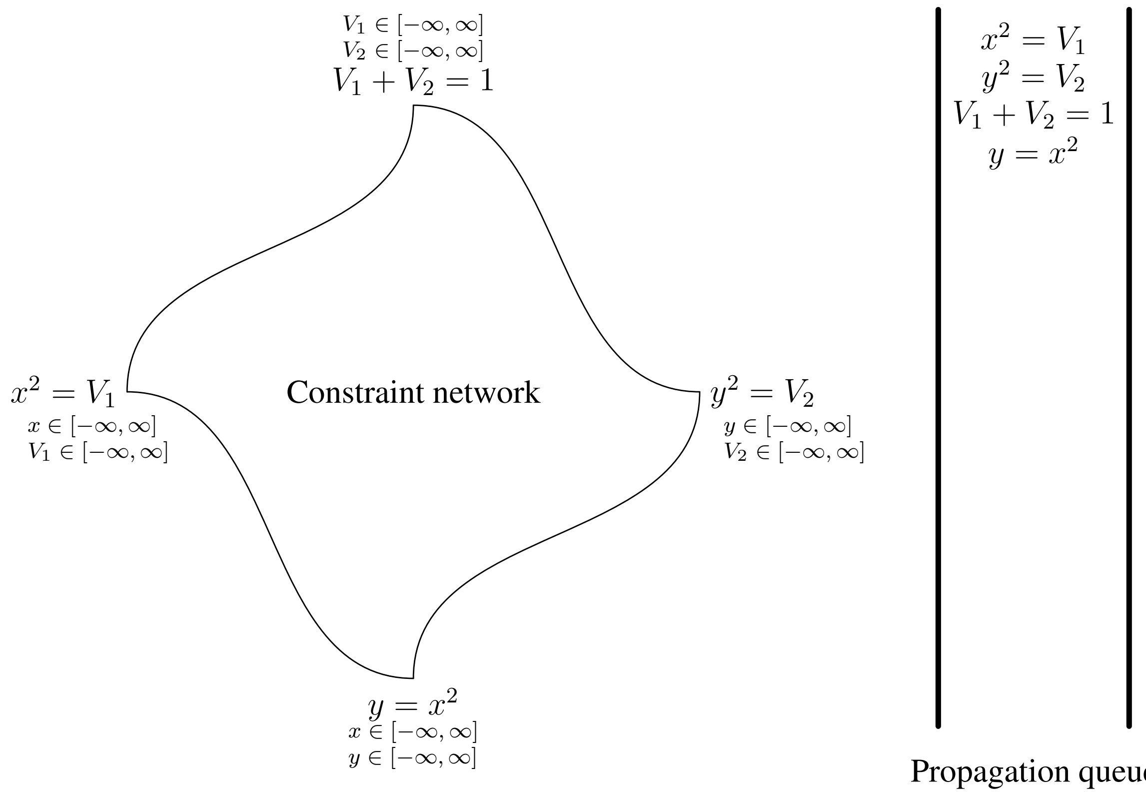
BNR Prolog system:

- Prolog implementation incorporating Cleary's ideas
- All operators are relations
- Interval arithmetic used to solve continuous constraints

$$\begin{cases} x^2 + y^2 = 1 \\ x^2 = y \\ x \in [-\infty, +\infty],\ y \in [-\infty, +\infty] \end{cases}$$

$$\downarrow$$

$$\begin{cases} x^2 = V_1 \\ y^2 = V_2 \\ V_1 + V_2 = 1 \\ y = x^2 \\ x \in [-\infty, +\infty],\ y \in [-\infty, +\infty] \\ V_1 \in [-\infty, +\infty],\ V_2 \in [-\infty, +\infty] \end{cases}$$



BNR Prolog [Older, 1989]

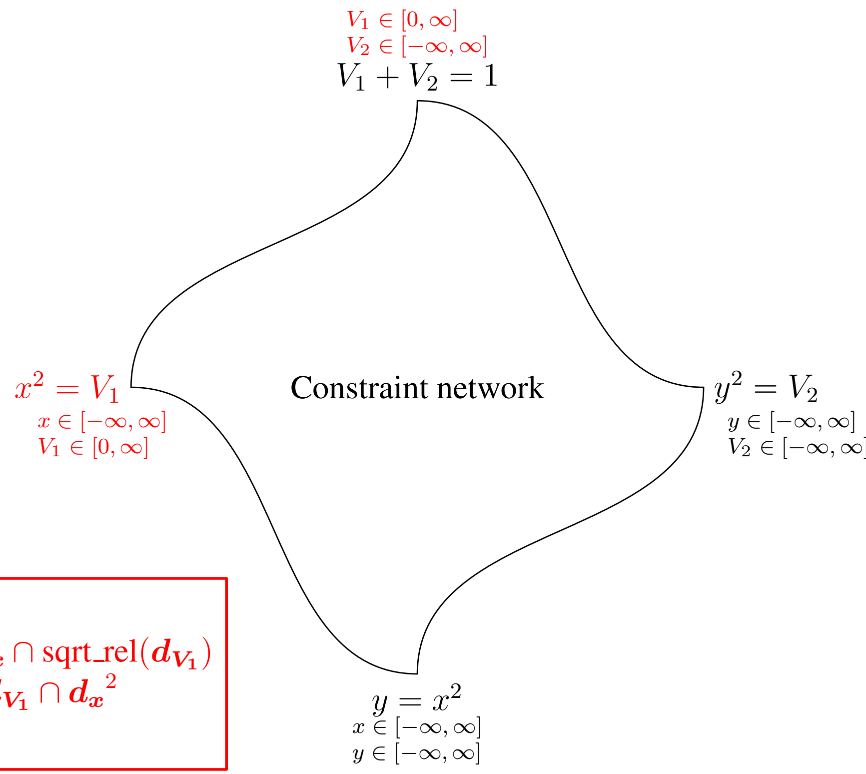
BNR Prolog system:

- Prolog implementation incorporating Cleary's ideas
- All operators are relations
- Interval arithmetic used to solve continuous constraints

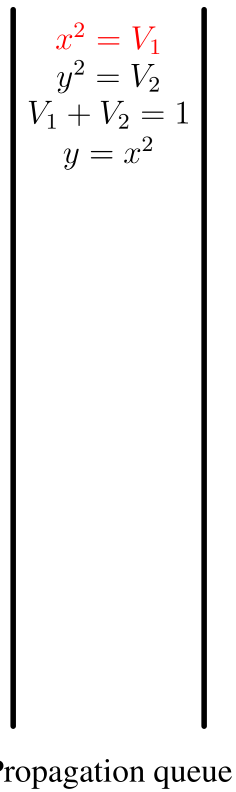
$$\begin{cases} x^2 + y^2 = 1 \\ x^2 = y \\ x \in [-\infty, +\infty],\ y \in [-\infty, +\infty] \end{cases}$$

$$\downarrow$$

$$\begin{cases} x^2 = V_1 \\ y^2 = V_2 \\ V_1 + V_2 = 1 \\ y = x^2 \\ x \in [-\infty, +\infty],\ y \in [-\infty, +\infty] \\ V_1 \in [-\infty, +\infty],\ V_2 \in [-\infty, +\infty] \end{cases}$$



$$\begin{cases} \boldsymbol{d_x} \leftarrow \boldsymbol{d_x} \cap \mathrm{sqrt\_rel}(\boldsymbol{d_{V_1}}) \\ \boldsymbol{d_{V_1}} \leftarrow \boldsymbol{d_{V_1}} \cap \boldsymbol{d_x}^2 \end{cases}$$



BNR Prolog [Older, 1989]

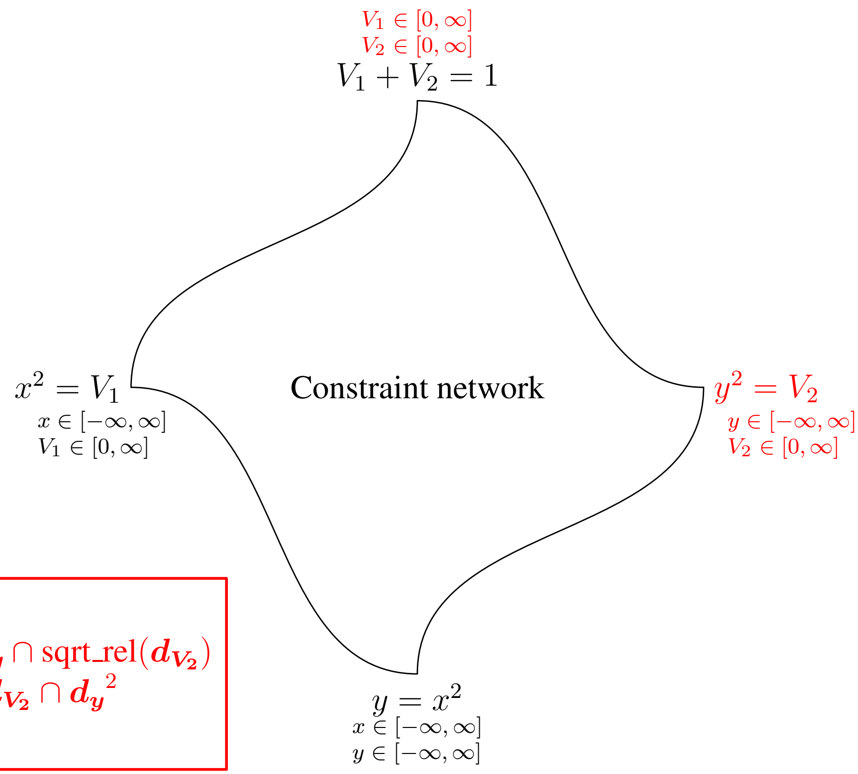
BNR Prolog system:

- Prolog implementation incorporating Cleary's ideas
- All operators are relations
- Interval arithmetic used to solve continuous constraints

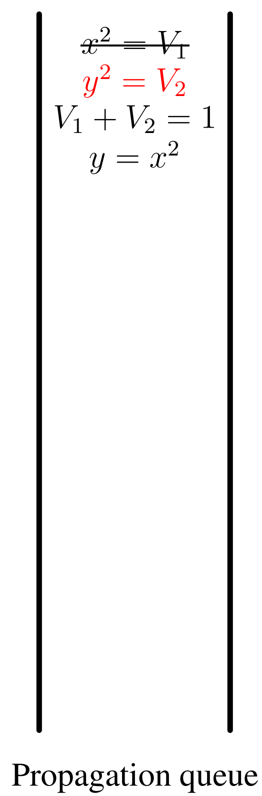
$$\begin{cases} x^2 + y^2 = 1 \\ x^2 = y \\ x \in [-\infty, +\infty],\ y \in [-\infty, +\infty] \end{cases}$$

$$\downarrow$$

$$\begin{cases} x^2 = V_1 \\ y^2 = V_2 \\ V_1 + V_2 = 1 \\ y = x^2 \\ x \in [-\infty, +\infty],\ y \in [-\infty, +\infty] \\ V_1 \in [-\infty, +\infty],\ V_2 \in [-\infty, +\infty] \end{cases}$$



$$\begin{cases} \boldsymbol{d_y} \leftarrow \boldsymbol{d_y} \cap \text{sqrt\_rel}(\boldsymbol{d_{V_2}}) \\ \boldsymbol{d_{V_2}} \leftarrow \boldsymbol{d_{V_2}} \cap \boldsymbol{d_y}^2 \end{cases}$$



BNR Prolog [Older, 1989]

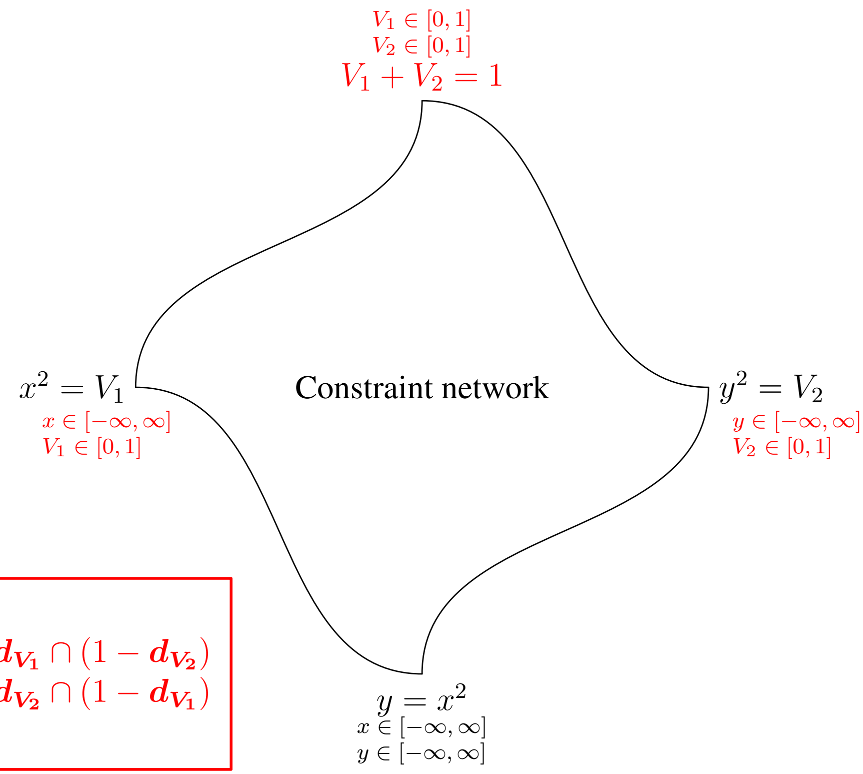
BNR Prolog system:

- Prolog implementation incorporating Cleary's ideas
- All operators are relations
- Interval arithmetic used to solve continuous constraints

$$\begin{cases} x^2 + y^2 = 1 \\ x^2 = y \\ x \in [-\infty, +\infty],\ y \in [-\infty, +\infty] \end{cases}$$

$$\downarrow$$

$$\begin{cases} x^2 = V_1 \\ y^2 = V_2 \\ V_1 + V_2 = 1 \\ y = x^2 \\ x \in [-\infty, +\infty],\ y \in [-\infty, +\infty] \\ V_1 \in [-\infty, +\infty],\ V_2 \in [-\infty, +\infty] \end{cases}$$



$$\begin{cases} \boldsymbol{d_{V_1}} \leftarrow \boldsymbol{d_{V_1}} \cap (1 - \boldsymbol{d_{V_2}}) \\ \boldsymbol{d_{V_2}} \leftarrow \boldsymbol{d_{V_2}} \cap (1 - \boldsymbol{d_{V_1}}) \end{cases}$$

~~$x^2 = V_1$~~
~~$y^2 = V_2$~~
$V_1 + V_2 = 1$
$y = x^2$

Propagation queue

BNR Prolog [Older, 1989]

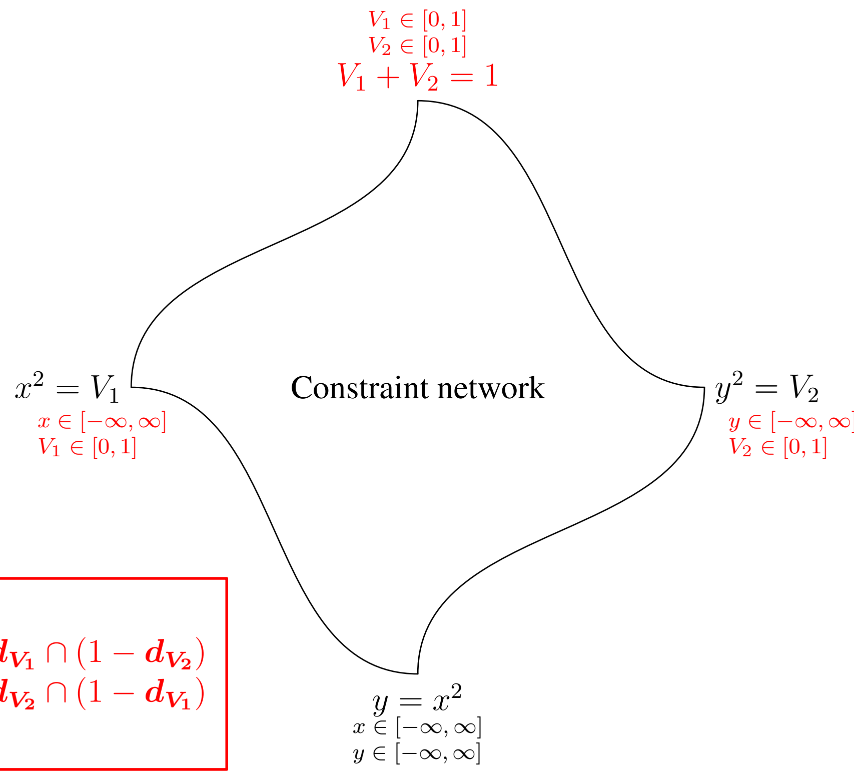
BNR Prolog system:

- Prolog implementation incorporating Cleary's ideas
- All operators are relations
- Interval arithmetic used to solve continuous constraints

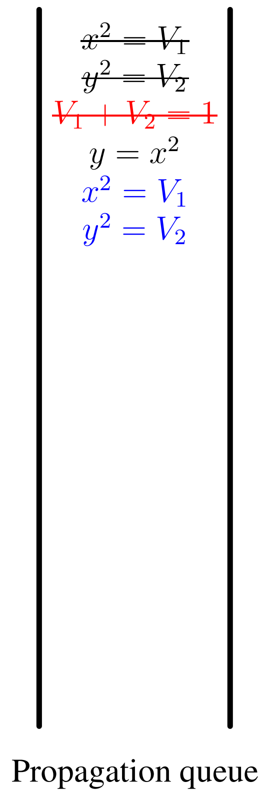
$$\begin{cases} x^2 + y^2 = 1 \\ x^2 = y \\ x \in [-\infty, +\infty],\ y \in [-\infty, +\infty] \end{cases}$$

$$\downarrow$$

$$\begin{cases} x^2 = V_1 \\ y^2 = V_2 \\ V_1 + V_2 = 1 \\ y = x^2 \\ x \in [-\infty, +\infty],\ y \in [-\infty, +\infty] \\ V_1 \in [-\infty, +\infty],\ V_2 \in [-\infty, +\infty] \end{cases}$$



$$\begin{cases} \boldsymbol{d_{V_1}} \leftarrow \boldsymbol{d_{V_1}} \cap (1 - \boldsymbol{d_{V_2}}) \\ \boldsymbol{d_{V_2}} \leftarrow \boldsymbol{d_{V_2}} \cap (1 - \boldsymbol{d_{V_1}}) \end{cases}$$



BNR Prolog [Older, 1989]

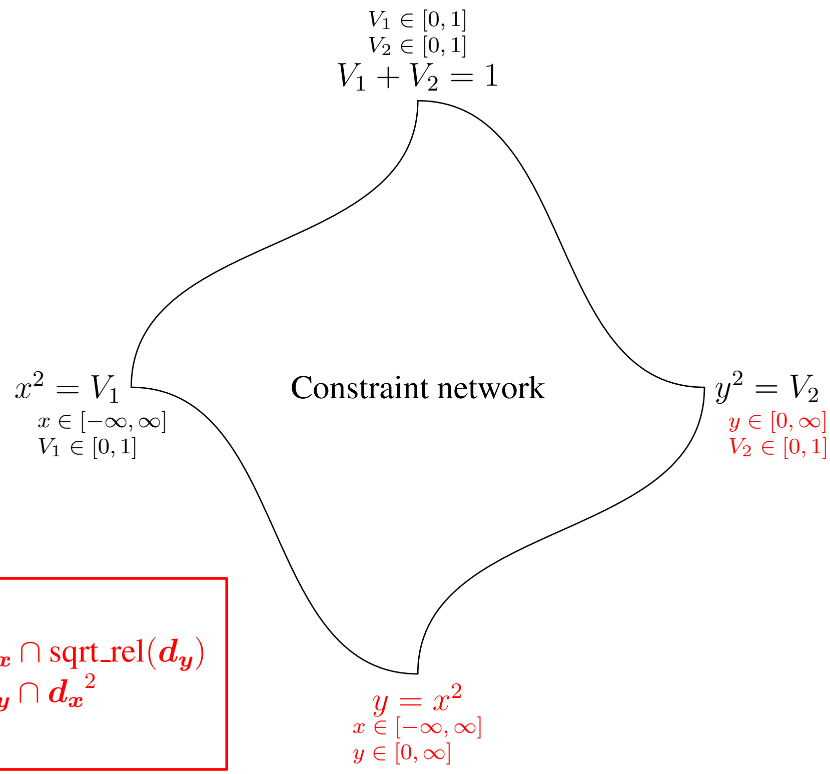
BNR Prolog system:

- Prolog implementation incorporating Cleary's ideas
- All operators are relations
- Interval arithmetic used to solve continuous constraints

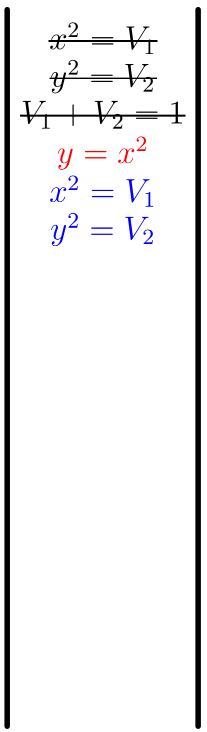
$$\begin{cases} x^2 + y^2 = 1 \\ x^2 = y \\ x \in [-\infty, +\infty],\ y \in [-\infty, +\infty] \end{cases}$$

$$\downarrow$$

$$\begin{cases} x^2 = V_1 \\ y^2 = V_2 \\ V_1 + V_2 = 1 \\ y = x^2 \\ x \in [-\infty, +\infty],\ y \in [-\infty, +\infty] \\ V_1 \in [-\infty, +\infty],\ V_2 \in [-\infty, +\infty] \end{cases}$$



$$\begin{cases} \boldsymbol{d_x} \leftarrow \boldsymbol{d_x} \cap \mathrm{sqrt\_rel}(\boldsymbol{d_y}) \\ \boldsymbol{d_y} \leftarrow \boldsymbol{d_y} \cap \boldsymbol{d_x}^2 \end{cases}$$



Propagation queue

BNR Prolog [Older, 1989]

BNR Prolog system:

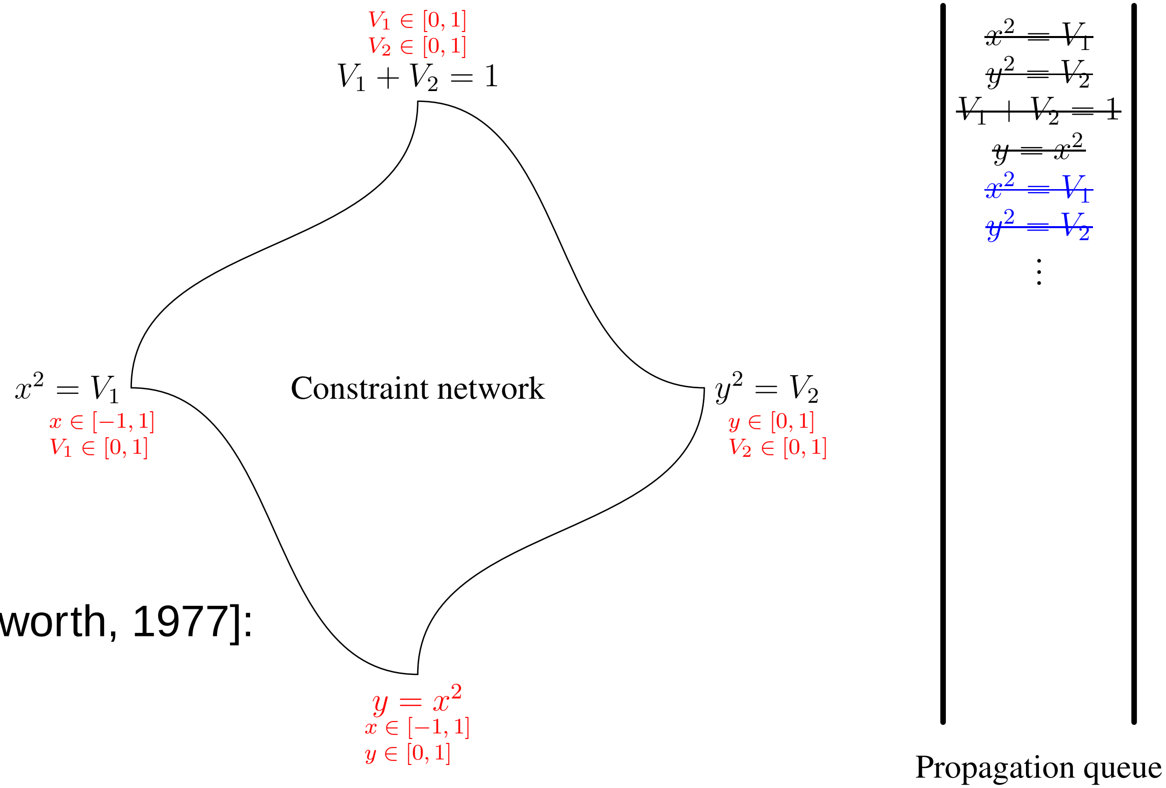
- Prolog implementation incorporating Cleary's ideas
- All operators are relations
- Interval arithmetic used to solve continuous constraints

$$\begin{cases} x^2 + y^2 = 1 \\ x^2 = y \\ x \in [-\infty, +\infty],\ y \in [-\infty, +\infty] \end{cases}$$

$$\downarrow$$

$$\begin{cases} x^2 = V_1 \\ y^2 = V_2 \\ V_1 + V_2 = 1 \\ y = x^2 \\ x \in [-\infty, +\infty],\ y \in [-\infty, +\infty] \\ V_1 \in [-\infty, +\infty],\ V_2 \in [-\infty, +\infty] \end{cases}$$

Domain reduction by primitives
$+$ intelligent propagation [Mackworth, 1977]:
Compare with [Kearfott, 1991]



Free-Steering Nonlinear Gauss-Seidel

GGS(**in** $F = (f_1, \ldots, f_n) \colon \mathbb{R}^n \rightarrow \mathbb{R}^n$; **inout** $\boldsymbol{B} = \boldsymbol{d_1} \times \cdots \times \boldsymbol{d_n} \in \mathbb{I}^n$)
begin
  modified $\leftarrow$ **true**;
  $\boldsymbol{B'} \leftarrow [-\infty, +\infty]^n$
  while w$(\boldsymbol{B}) > \varepsilon$ **and modified do**
    Lfv $\leftarrow$ select$(\{f_1, \ldots, f_n\}, \{x_1, \ldots, x_n\}, \boldsymbol{B'}, \boldsymbol{B})$
    $\boldsymbol{B'} \leftarrow \boldsymbol{B}$
    foreach (f_i, v_j) **in Lfv do**
      $\boldsymbol{d_j} \leftarrow \boldsymbol{d_j} \cap$ tighten$(f_i, v_j, \boldsymbol{B})$
    endfor
    modified $\leftarrow$ (dist$(\boldsymbol{B}, \boldsymbol{B'}) > \Delta$)
  endwhile
end

Hull consistency

[Benhamou and Older, 1997]: Formalization of BNR Prolog algorithms

Definition 1 *A constraint* $c(x_1, \ldots, x_n)$ *is* hull consistent *w.r.t. a box* $\boldsymbol{B} = \boldsymbol{d_1} \times \cdots \times \boldsymbol{d_n}$ *iff* $\boldsymbol{B}$ *cannot be tightened without losing solutions of* c*.*

Example: $c\colon x + y = z$ is not hull consistent w.r.t. $([3, 5], [-2, 4], [0, 9])$.

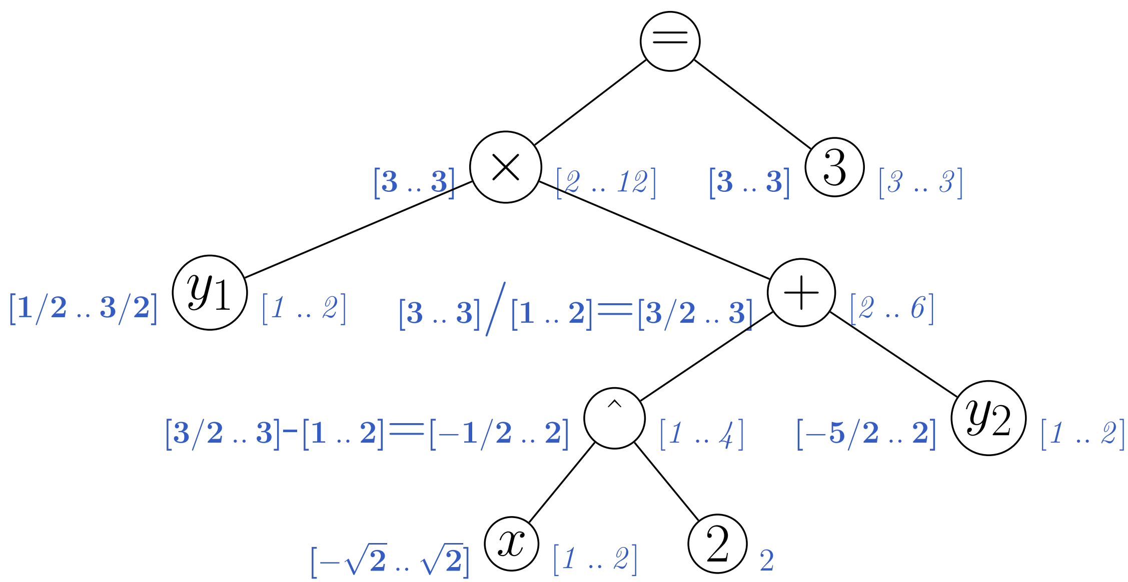
- Hull consistency "easy" to compute for *primitives* ($xy = z$, $x + y = z$, $x^n = y$, $\cos(x) = y$,...)
- Computing the tightess box for arbitrary constraints is difficult

HC3

1. Decomposition of a constraint c into a conjunction of primitives $c_1 \wedge \cdots \wedge c_p$
2. **Definition:** A conjunction of constraints $c_1 \wedge \cdots \wedge c_p$ is hull consistent w.r.t. a box $\boldsymbol{B}$ iff c_1, ..., c_p are hull consistent w.r.t. $\boldsymbol{B}$
3. Efficient propagation in constraint network by AC3-like algorithm [Mackworth, 1977]

Avoiding decomposition

- **HC3** inefficient for large constraint systems with “complicated” constraints
- **HC4** [Benhamou et al., 1999]: bottom-up and top-down sweep in constraint expression tree (no decomposition)
  HC4 on $y(x^2 + y) = 3$, $\boldsymbol{d_x} = [1, 2]$, $\boldsymbol{d_y} = [1, 2]$?

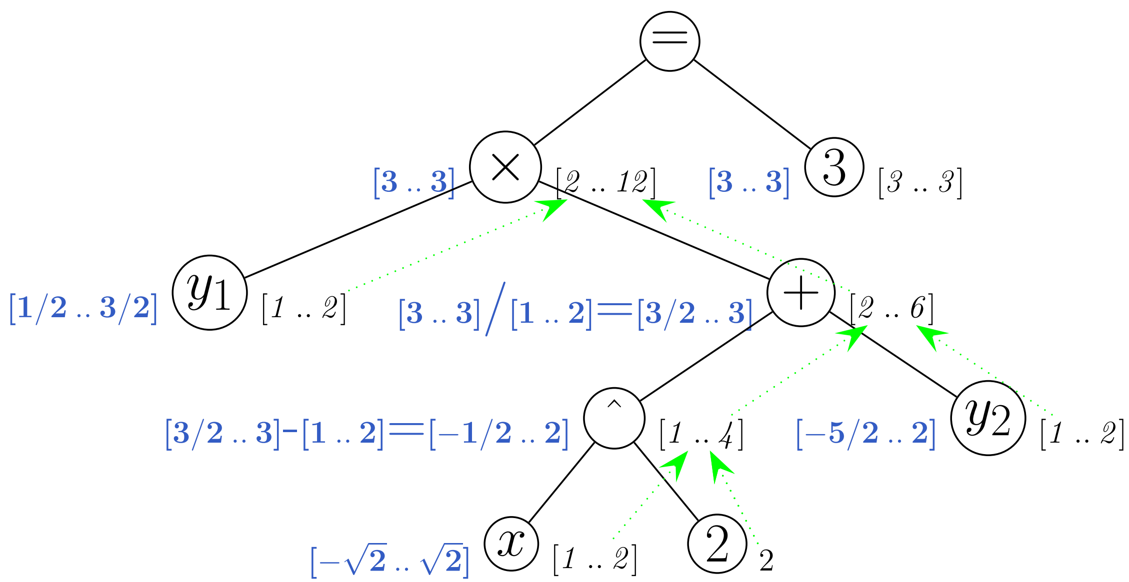


Two sweeps in the tree:

$$\begin{cases} \boldsymbol{d'_x} \leftarrow \boldsymbol{d_x} \cap \sqrt{\frac{[3,3]}{\boldsymbol{d_{y_1}}} - \boldsymbol{d_{y_2}}} \\ \boldsymbol{d'_{y_1}} \leftarrow \boldsymbol{d_{y_1}} \cap \frac{[3,3]}{\boldsymbol{d_x}^2 + \boldsymbol{d_{y_2}}} \\ \boldsymbol{d'_{y_2}} \leftarrow \boldsymbol{d_{y_2}} \cap \left(\frac{[3,3]}{\boldsymbol{d_{y_1}}} - \boldsymbol{d_x}^2\right) \end{cases}$$

Avoiding decomposition

- **HC3** inefficient for large constraint systems with "complicated" constraints
- **HC4** [Benhamou et al., 1999]: bottom-up and top-down sweep in constraint expression tree (no decomposition)
  HC4 on $y(x^2 + y) = 3$, $\boldsymbol{d_x} = [1, 2]$, $\boldsymbol{d_y} = [1, 2]$?

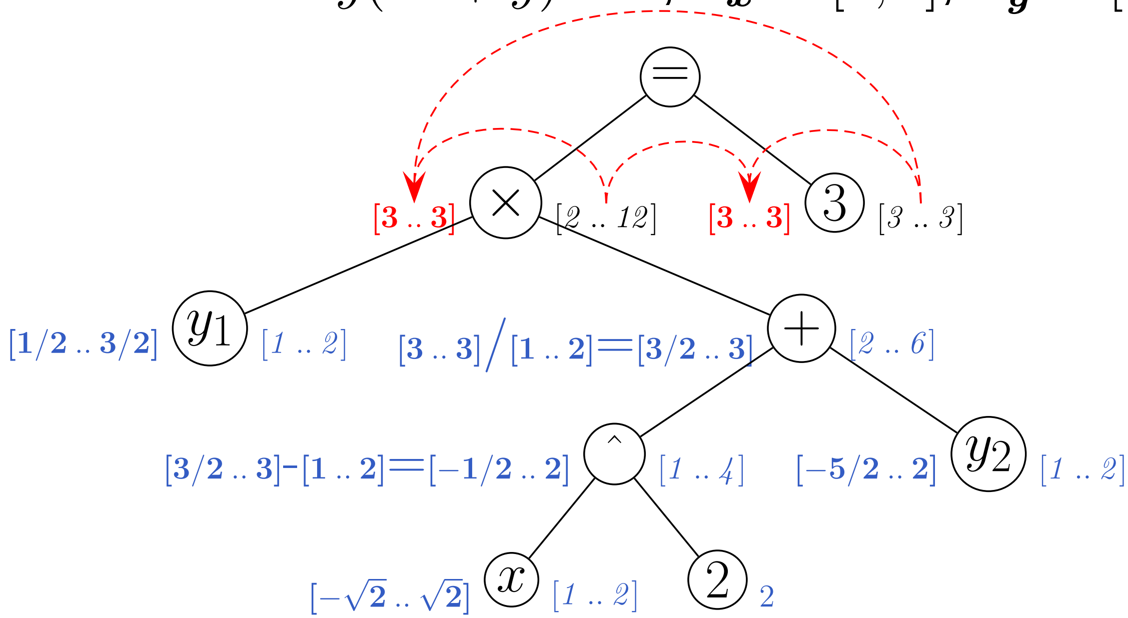


Two sweeps in the tree:

$$\begin{cases} \boldsymbol{d'_x} \leftarrow \boldsymbol{d_x} \cap \sqrt{\frac{[3,3]}{\boldsymbol{d_{y_1}}} - \boldsymbol{d_{y_2}}} \\ \boldsymbol{d'_{y_1}} \leftarrow \boldsymbol{d_{y_1}} \cap \frac{[3,3]}{\boldsymbol{d_x}^2 + \boldsymbol{d_{y_2}}} \\ \boldsymbol{d'_{y_2}} \leftarrow \boldsymbol{d_{y_2}} \cap \left(\frac{[3,3]}{\boldsymbol{d_{y_1}}} - \boldsymbol{d_x}^2\right) \end{cases}$$

Avoiding decomposition

- **HC3** inefficient for large constraint systems with “complicated” constraints
- **HC4** [Benhamou et al., 1999]: bottom-up and top-down sweep in constraint expression tree (no decomposition)
  HC4 on $y(x^2+y)=3$, $\boldsymbol{d_x}=[1,2]$, $\boldsymbol{d_y}=[1,2]$?



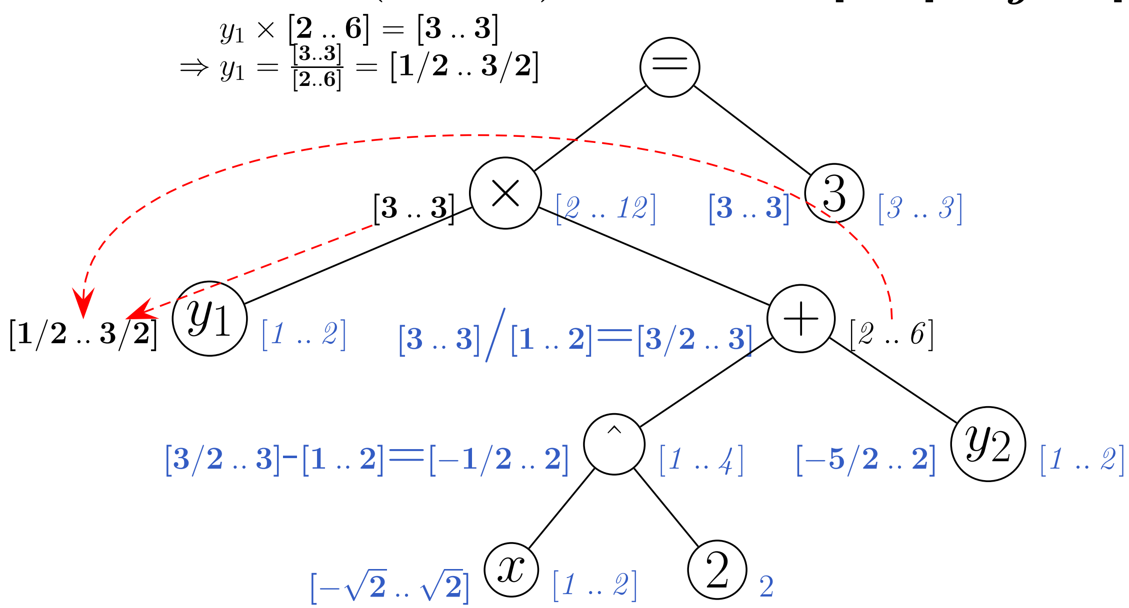
Two sweeps in the tree:

$$\begin{cases} \boldsymbol{d'_x} \leftarrow \boldsymbol{d_x} \cap \sqrt{\frac{[3,3]}{\boldsymbol{d_{y_1}}} - \boldsymbol{d_{y_2}}} \\ \boldsymbol{d'_{y_1}} \leftarrow \boldsymbol{d_{y_1}} \cap \frac{[3,3]}{\boldsymbol{d_x}^2 + \boldsymbol{d_{y_2}}} \\ \boldsymbol{d'_{y_2}} \leftarrow \boldsymbol{d_{y_2}} \cap \left(\frac{[3,3]}{\boldsymbol{d_{y_1}}} - \boldsymbol{d_x}^2\right) \end{cases}$$

Avoiding decomposition

- **HC3** inefficient for large constraint systems with “complicated” constraints
- **HC4** [Benhamou et al., 1999]: bottom-up and top-down sweep in constraint expression tree (no decomposition)

  HC4 on $y(x^2+y)=3$, $\boldsymbol{d_x}=[1,2]$, $\boldsymbol{d_y}=[1,2]$?

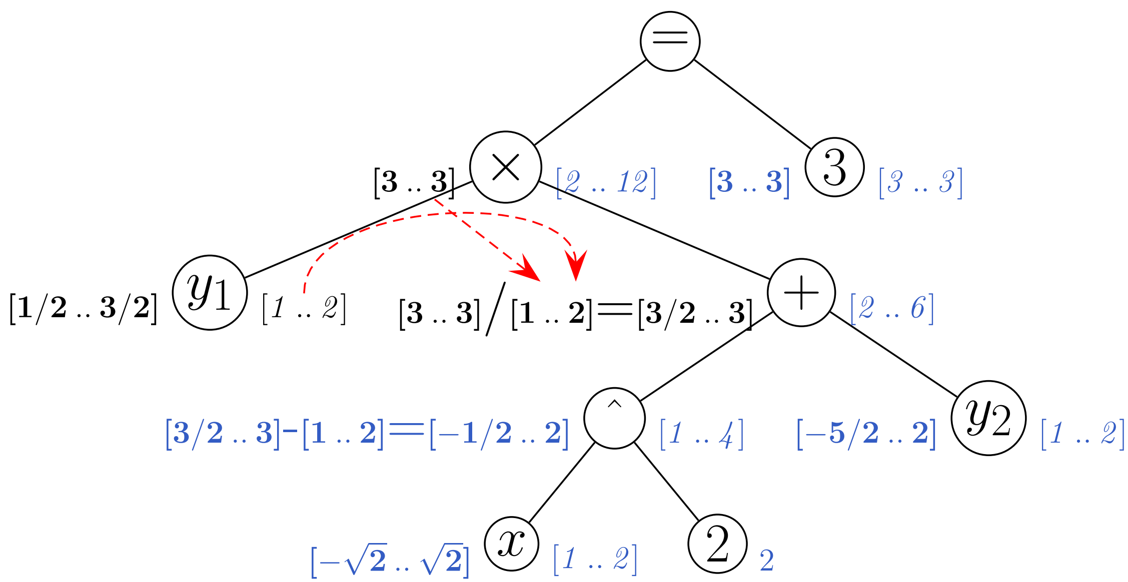


Two sweeps in the tree:

$$\begin{cases} \boldsymbol{d'_x} \leftarrow \boldsymbol{d_x} \cap \sqrt{\frac{[3,3]}{\boldsymbol{d_{y_1}}} - \boldsymbol{d_{y_2}}} \\ \boldsymbol{d'_{y_1}} \leftarrow \boldsymbol{d_{y_1}} \cap \frac{[3,3]}{\boldsymbol{d_x}^2 + \boldsymbol{d_{y_2}}} \\ \boldsymbol{d'_{y_2}} \leftarrow \boldsymbol{d_{y_2}} \cap \left(\frac{[3,3]}{\boldsymbol{d_{y_1}}} - \boldsymbol{d_x}^2\right) \end{cases}$$

Avoiding decomposition

- **HC3** inefficient for large constraint systems with “complicated” constraints
- **HC4** [Benhamou et al., 1999]: bottom-up and top-down sweep in constraint expression tree (no decomposition)
  HC4 on $y(x^2+y)=3$, $\boldsymbol{d_x}=[1,2]$, $\boldsymbol{d_y}=[1,2]$?

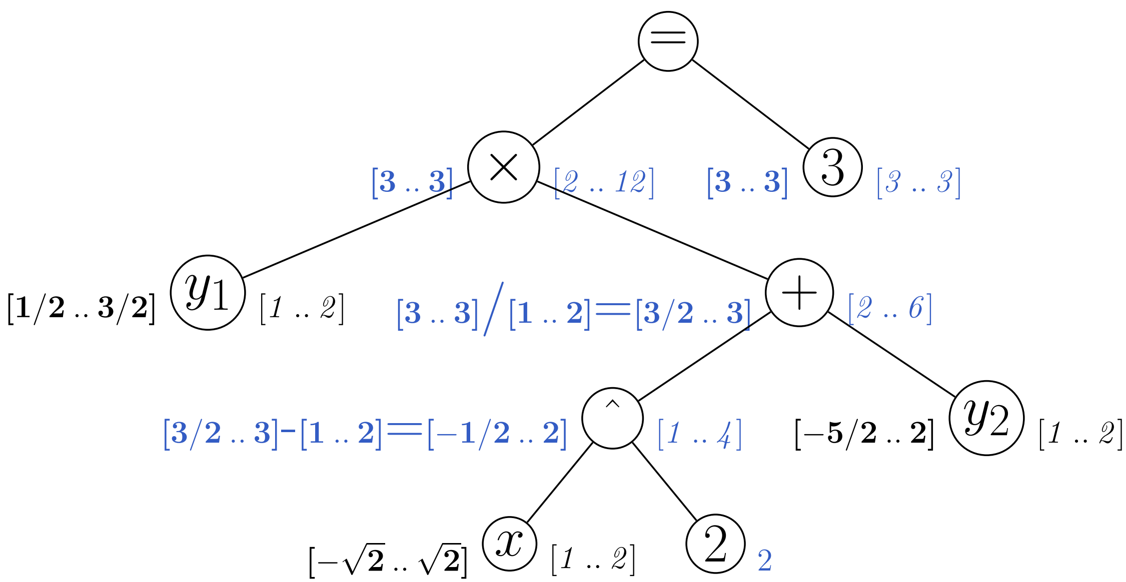


Two sweeps in the tree:

$$\begin{cases} \boldsymbol{d'_x} \leftarrow \boldsymbol{d_x} \cap \sqrt{\frac{[3,3]}{\boldsymbol{d_{y_1}}} - \boldsymbol{d_{y_2}}} \\ \boldsymbol{d'_{y_1}} \leftarrow \boldsymbol{d_{y_1}} \cap \frac{[3,3]}{\boldsymbol{d_x}^2 + \boldsymbol{d_{y_2}}} \\ \boldsymbol{d'_{y_2}} \leftarrow \boldsymbol{d_{y_2}} \cap \left(\frac{[3,3]}{\boldsymbol{d_{y_1}}} - \boldsymbol{d_x}^2\right) \end{cases}$$

Avoiding decomposition

- **HC3** inefficient for large constraint systems with “complicated” constraints
- **HC4** [Benhamou et al., 1999]: bottom-up and top-down sweep in constraint expression tree (no decomposition)
  HC4 on $y(x^2+y)=3$, $\boldsymbol{d_x}=[1,2]$, $\boldsymbol{d_y}=[1,2]$?



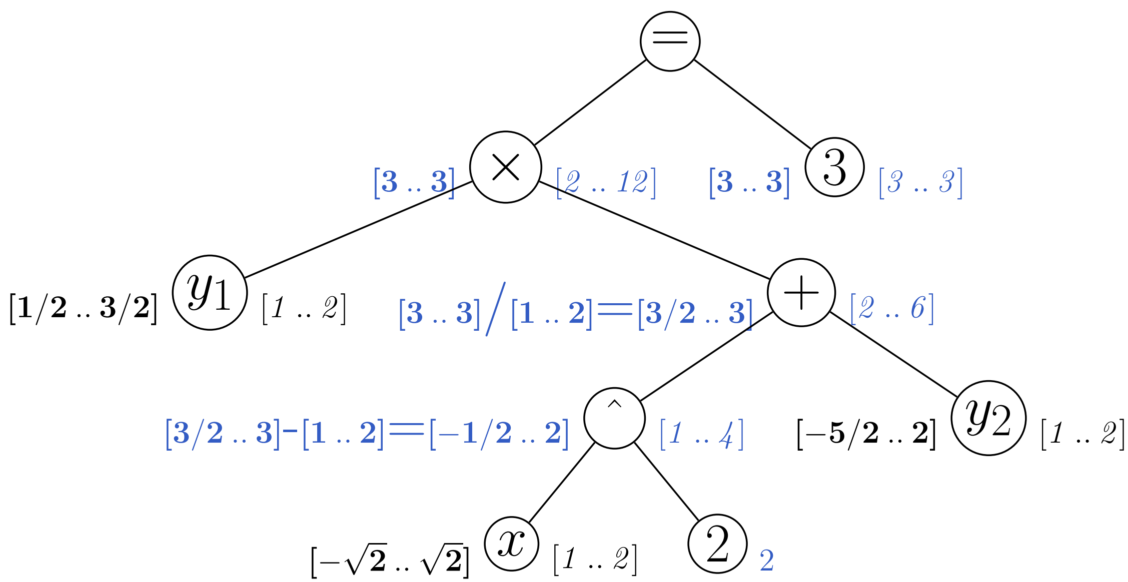
Two sweeps in the tree:

$$\begin{cases} \boldsymbol{d'_x} \leftarrow \boldsymbol{d_x} \cap \sqrt{\frac{[3,3]}{\boldsymbol{d_{y_1}}} - \boldsymbol{d_{y_2}}} \\ \boldsymbol{d'_{y_1}} \leftarrow \boldsymbol{d_{y_1}} \cap \frac{[3,3]}{\boldsymbol{d_x}^2 + \boldsymbol{d_{y_2}}} \\ \boldsymbol{d'_{y_2}} \leftarrow \boldsymbol{d_{y_2}} \cap \left(\frac{[3,3]}{\boldsymbol{d_{y_1}}} - \boldsymbol{d_x}^2\right) \end{cases}$$

HC4revise:
$$\begin{cases} \boldsymbol{d''_y} \leftarrow \boldsymbol{d_y} \cap \boldsymbol{d'_{y_1}} \cap \boldsymbol{d'_{y_2}} = [1, 3/2] \\ \boldsymbol{d''_x} \leftarrow \boldsymbol{d_x} \cap \boldsymbol{d'_x} = [1, \sqrt{2}] \end{cases}$$

Avoiding decomposition

- **HC3** inefficient for large constraint systems with “complicated” constraints
- **HC4** [Benhamou et al., 1999]: bottom-up and top-down sweep in constraint expression tree (no decomposition)
  HC4 on $y(x^2+y)=3$, $\boldsymbol{d_x}=[1,2]$, $\boldsymbol{d_y}=[1,2]$?



Two sweeps in the tree:

$$\begin{cases} \boldsymbol{d'_x} \leftarrow \boldsymbol{d_x} \cap \sqrt{\frac{[3,3]}{\boldsymbol{d_{y_1}}} - \boldsymbol{d_{y_2}}} \\ \boldsymbol{d'_{y_1}} \leftarrow \boldsymbol{d_{y_1}} \cap \frac{[3,3]}{\boldsymbol{d_x}^2 + \boldsymbol{d_{y_2}}} \\ \boldsymbol{d'_{y_2}} \leftarrow \boldsymbol{d_{y_2}} \cap \left(\frac{[3,3]}{\boldsymbol{d_{y_1}}} - \boldsymbol{d_x}^2\right) \end{cases}$$

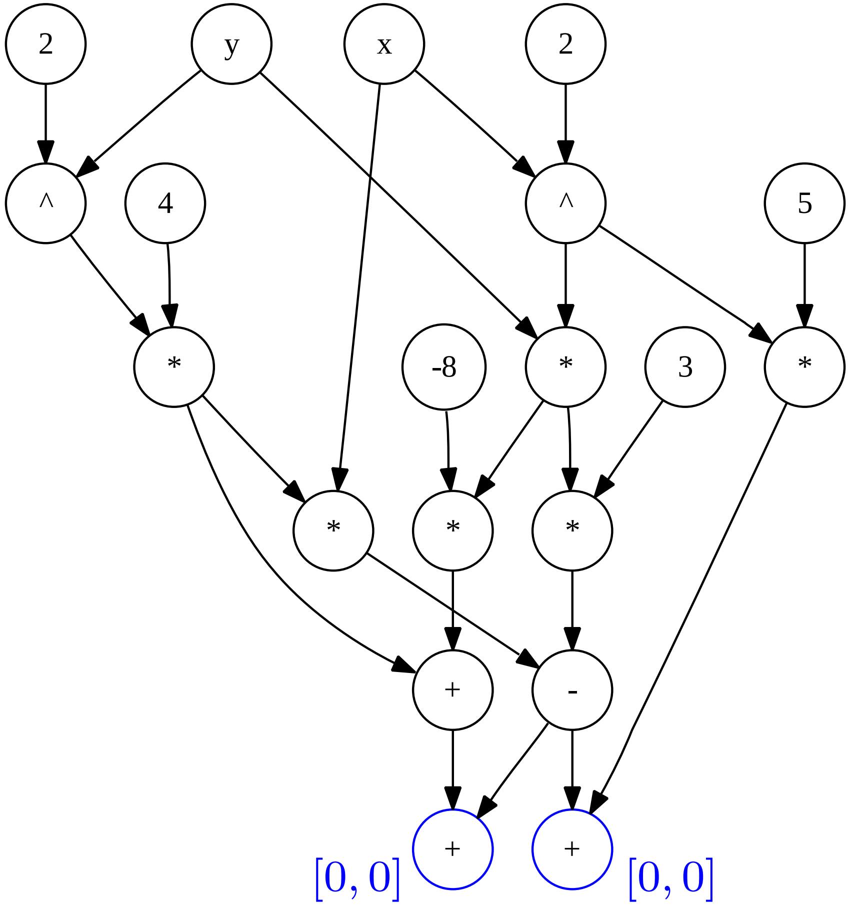
HC4 is in essence [Kearfott, 1991]’s Alg. 3.1 with particular ordering

Even faster: FBPD [Vu et al., 2004]

Vu, Schichl, & Sam-Haroud:

- Propagation on DAGs instead of trees
- Propagation on selected variables

$$\begin{cases} 3x^2y - 4xy^2 + 5x^2 & = 0 \\ -8x^2y + 4y^2 - 4xy^2 & = 0 \end{cases}$$



Effective use of relational operators

- Relational operators make for cheap contracting algorithms (e.g., HC4)
- Good reduction of large boxes (see circle/parabola example)
- HC4-like algorithms at their best if no multi-occurrence of variables

[Granvilliers and Benhamou, 2001] : smart combination of multidimensional interval Newton method and HC4 to solve the Ebers & Moll circuit design problem
[Ebers and Moll, 1954]

Note: in general, HC4 [Benhamou et al., 1999] and HC [Hansen and Walster, 2003] do *not* achieve hull consistency

Availability of relational arithmetic

- **Tools**

  Realpaver : constraint solver with HC3, HC4, and more

  Globsol : HC3/HC4-like

  ECLIPSe : HC3

  Prolog IV : HC3

- **Libraries**

  Ilog Solver : HC3, HC4? (operators directly accessible?)

  smath : almost all relational operators in a C library

  Boost : unreleased as of v. 1.34.1

  gaol : relational operators specified by the C++ standard proposal [Brönnimann et al., 2006] (except `atan2_rel`)

Gaol

Gaol is not **J**ust **A**nother **I**nterval **L**ibrary

- C++ library developed at EPFL, Switzerland from 2000 to 2001 and in Nantes, France since 2001
- Availability from SourceForge: `http://sf.net/projects/gaol/`
- Implements functional as well as relational interval arithmetic
- New version (not yet released) uses SIMD SSE2 instructions
- Algorithms detailed in Technical Report hal-00288457
  Interval Extensions of Multivalued Inverse Functions, F. Goualard, 2007
- Used in *Constraint Explorer* (Dassault Aviation), and research projects at various universities and labs. [CWI, (UTEP?), UNantes, UMelbourne. . .]

Relational arithmetic & standards

- C++ Standard Library proposal
  Relational operators available
  (Section 26.6.15 **mathematical relations**):
  - acos_rel$(\boldsymbol{d_x}, \boldsymbol{d_r}) = \mathrm{cch}\,\{r \in \boldsymbol{d_r} \mid \cos r \in \boldsymbol{d_x}\}$
  - acosh_rel$(\boldsymbol{d_x}, \boldsymbol{d_r}) = \mathrm{cch}\,\{r \in \boldsymbol{d_r} \mid \cosh r \in \boldsymbol{d_x}\}$
  - asin_rel$(\boldsymbol{d_x}, \boldsymbol{d_r}) = \mathrm{cch}\,\{r \in \boldsymbol{d_r} \mid \sin r \in \boldsymbol{d_x}\}$
  - atan_rel$(\boldsymbol{d_x}, \boldsymbol{d_r}) = \mathrm{cch}\,\{r \in \boldsymbol{d_r} \mid \tan r \in \boldsymbol{d_x}\}$
  - atan2_rel$(\boldsymbol{d_y}, \boldsymbol{d_x}, \boldsymbol{d_r}) = \mathrm{cch}\,\{r \in \boldsymbol{d_r} \mid \cos r \in \boldsymbol{d_x} \wedge \sin r \in \boldsymbol{d_y}\}$
  - sqrt_rel$(\boldsymbol{d_x}, \boldsymbol{d_r}) = \mathrm{cch}\,\{r \in \boldsymbol{d_r} \mid r^2 \in \boldsymbol{d_x}\}$
  - nth_root_rel$(\boldsymbol{d_x}, n, \boldsymbol{d_r}) = \mathrm{cch}\,\{r \in \boldsymbol{d_r} \mid r^n \in \boldsymbol{d_x}\}$
  - div_rel$(\boldsymbol{d_x}, \boldsymbol{d_y}) = \mathrm{cch}\,\{r \in \mathbb{R} \mid \exists y \in \boldsymbol{d_y} \colon ry \in \boldsymbol{d_x}\}$
    In gaol: div_rel$(\boldsymbol{d_x}, \boldsymbol{d_y}, \boldsymbol{d_r}) = \mathrm{cch}\,\{r \in \boldsymbol{d_r} \mid \exists y \in \boldsymbol{d_y} \colon ry \in \boldsymbol{d_x}\}$

Relational arithmetic & standards

- C++ Standard Library proposal
  Relational operators available
  (Section 26.6.15 **mathematical relations**):
  - acos_rel$(\boldsymbol{d_x}, \boldsymbol{d_r}) = \mathrm{cch}\,\{r \in \boldsymbol{d_r} \mid \cos r \in \boldsymbol{d_x}\}$
  - acosh_rel$(\boldsymbol{d_x}, \boldsymbol{d_r}) = \mathrm{cch}\,\{r \in \boldsymbol{d_r} \mid \cosh r \in \boldsymbol{d_x}\}$
  - asin_rel$(\boldsymbol{d_x}, \boldsymbol{d_r}) = \mathrm{cch}\,\{r \in \boldsymbol{d_r} \mid \sin r \in \boldsymbol{d_x}\}$
  - atan_rel$(\boldsymbol{d_x}, \boldsymbol{d_r}) = \mathrm{cch}\,\{r \in \boldsymbol{d_r} \mid \tan r \in \boldsymbol{d_x}\}$
  - atan2_rel$(\boldsymbol{d_y}, \boldsymbol{d_x}, \boldsymbol{d_r}) = \mathrm{cch}\,\{r \in \boldsymbol{d_r} \mid \cos r \in \boldsymbol{d_x} \wedge \sin r \in \boldsymbol{d_y}\}$
  - sqrt_rel$(\boldsymbol{d_x}, \boldsymbol{d_r}) = \mathrm{cch}\,\{r \in \boldsymbol{d_r} \mid r^2 \in \boldsymbol{d_x}\}$
  - nth_root_rel$(\boldsymbol{d_x}, n, \boldsymbol{d_r}) = \mathrm{cch}\,\{r \in \boldsymbol{d_r} \mid r^n \in \boldsymbol{d_x}\}$
  - div_rel$(\boldsymbol{d_x}, \boldsymbol{d_y}) = \mathrm{cch}\,\{r \in \mathbb{R} \mid \exists y \in \boldsymbol{d_y} \colon ry \in \boldsymbol{d_x}\}$
    In gaol: div_rel$(\boldsymbol{d_x}, \boldsymbol{d_y}, \boldsymbol{d_r}) = \mathrm{cch}\,\{r \in \boldsymbol{d_r} \mid \exists y \in \boldsymbol{d_y} \colon ry \in \boldsymbol{d_x}\}$
- What about IEEE Interval arithmetic standard?
  See Neumaier’s draft

Interval Multivalued Inverse Functions

Relational Interval Arithmetic and its Use

Frédéric Goualard

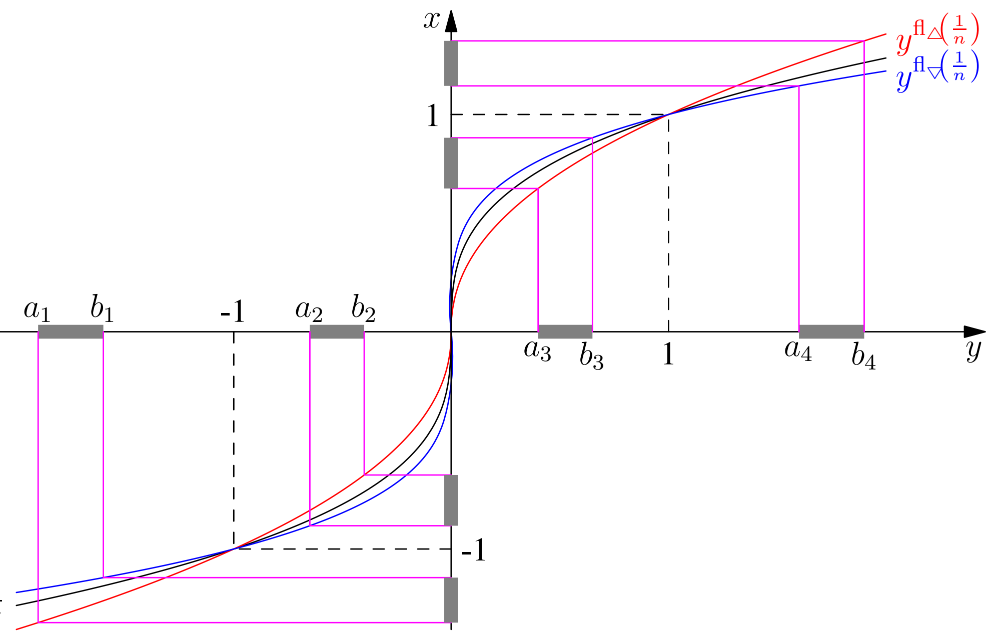
Frederic.Goualard@univ-nantes.fr

Laboratoire d’Informatique de Nantes-Atlantique, UMR CNRS 6241

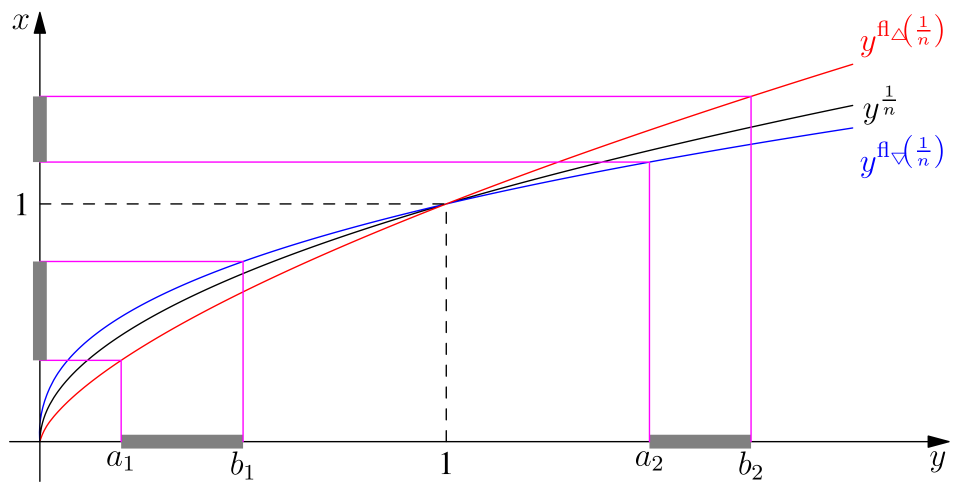
Implementation

$x^n = y$ (1)

$$\text{nth\_root\_rel}(\mathbf{d_y}, n, \mathbf{d_x}) = \text{cch}\,\{x \in \mathbf{d_x} \mid \exists y \in \mathbf{d_y} : y = x^n\}, \quad n \in \mathbb{N}.$$



Odd powers



Even powers

$x^n = y$ (2)

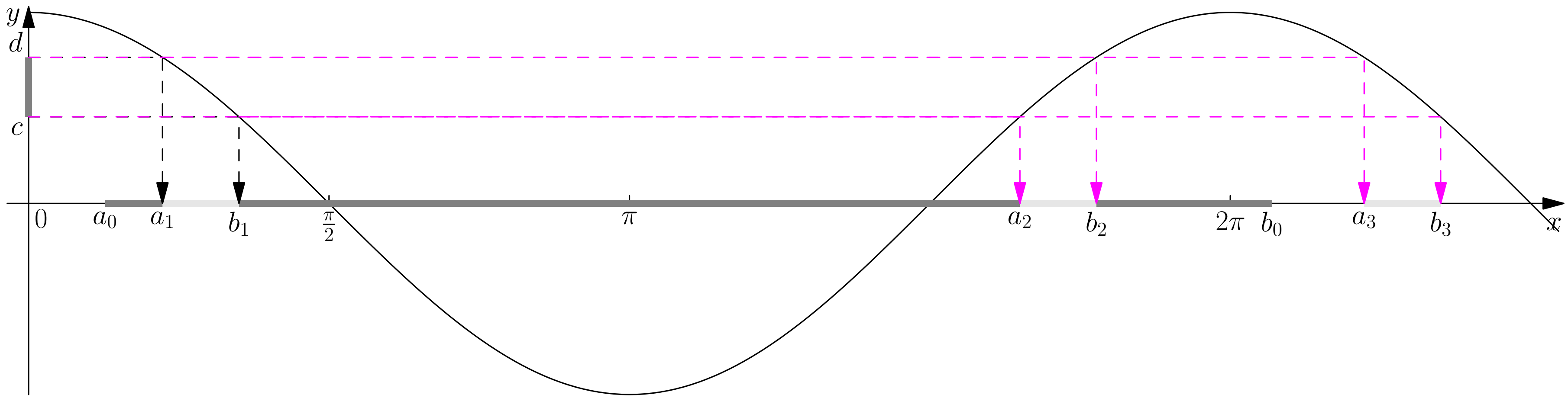
 # Computes an enclosing interval for
$\operatorname{cch}\{x \in \mathbf{d_x} \mid \exists y \in \mathbf{d_y} : y = x^n\}$

function **nth_root_rel**$(\mathbf{d_y} \in \mathbb{I}, n \in \mathbb{N}, \mathbf{d_x} \in \mathbb{I})$:

- **if** $n = 0$:
  - **if** $1 \in \mathbf{d_y}$:
    - **return** $\mathbf{d_x}$
  - **else**:
    - **return** $\varnothing$
- **elsif** $n = 1$:
  - **return** $\mathbf{d_x} \cap \mathbf{d_y}$
- **else**:
  - **if** **odd**(n):
    - **if** $\mathbf{d_x} = \varnothing \vee \mathbf{d_y} = \varnothing$:
      - **return** $\varnothing$
    - *# Computing the left bound*
    - **if** $\underline{\mathbf{d_y}} \geqslant 1$:
      - $l \leftarrow$ **pow_dn**$(\underline{\mathbf{d_y}}, O)$
    - **elsif** $\underline{\mathbf{d_y}} \geqslant 0$:
      - $l \leftarrow$ **pow_dn**$(\underline{\mathbf{d_y}}, \operatorname{fl}_\triangle\left(\frac{1}{n}\right))$
    - **elsif** $\underline{\mathbf{d_y}} \geqslant -1$:
      - $l \leftarrow$ **pow_dn**$(\underline{\mathbf{d_y}}, O)$
    - **else**:
      - $l \leftarrow$ **pow_dn**$(\underline{\mathbf{d_y}}, \operatorname{fl}_\triangle\left(\frac{1}{n}\right))$
    - *# Computing the right bound*
    - **if** $\overline{\mathbf{d_y}} \geqslant 1$:
      - $l \leftarrow$ **pow_up**$(\overline{\mathbf{d_y}}, \operatorname{fl}_\triangle\left(\frac{1}{n}\right))$
    - **elsif** $\overline{\mathbf{d_y}} \geqslant 0$:
      - $l \leftarrow$ **pow_up**$(\overline{\mathbf{d_y}}, O)$
    - **elsif** $\overline{\mathbf{d_y}} \geqslant -1$:
      - $l \leftarrow$ **pow_up**$(\overline{\mathbf{d_y}}, \operatorname{fl}_\triangle\left(\frac{1}{n}\right))$
    - **else**:
      - $l \leftarrow$ **pow_up**$(\overline{\mathbf{d_y}}, O)$
    - **return** $[l, r] \cap \mathbf{d_x}$
  - **else**: *# even*(n)
    - $\mathbf{d_y}' \leftarrow \mathbf{d_y} \cap [0, +\infty]$
    - **if** $\mathbf{d_x} = \varnothing \vee \mathbf{d_y}' = \varnothing$:
      - **return** $\varnothing$
    - *# Computing the left bound*
    - **if** $\underline{\mathbf{d_y}} \geqslant 1$:
      - $l \leftarrow$ **pow_dn**$(\underline{\mathbf{d_y}}, O)$
    - **else**:
      - $l \leftarrow$ **pow_dn**$(\underline{\mathbf{d_y}}, \operatorname{fl}_\triangle\left(\frac{1}{n}\right))$
    - *# Computing the right bound*
    - **if** $\overline{\mathbf{d_y}} \geqslant 1$:
      - $l \leftarrow$ **pow_up**$(\overline{\mathbf{d_y}}, \operatorname{fl}_\triangle\left(\frac{1}{n}\right))$
    - **else**:
      - $l \leftarrow$ **pow_up**$(\overline{\mathbf{d_y}}, O)$
    - **return** $\operatorname{cch}\left(([l, r] \cap \mathbf{d_x}) \cup ([-r, -l] \cap \mathbf{d_x})\right)$

$\cos(x) = y$ (1)

Computing the inverse cosine of $[c, d]$ w.r.t. $[a_0, b_0]$



$\cos(x) = y$ (2)

 # Returns an enclosing interval for the preimage of $\mathbf{d_y}$
 # w.r.t. the cosine function and $\mathbf{d_x}$:
 # acos_rel$(\mathbf{d_y}, \mathbf{d_x}) \supseteq \operatorname{cch}\{x \in \mathbf{d_x} \mid \exists y \in \mathbf{d_y} : y = \cos x\}$

function acos_rel$(\mathbf{d_y} \in \mathbb{I}, \mathbf{d_x} \in \mathbb{I})$:
- **if** $\mathbf{d_x} = \varnothing \vee (\mathbf{d_y} \cap [-1, 1]) = \varnothing$:
  - **return** $\varnothing$
- **if** $[-1, 1] \subseteq \mathbf{d_y}$:
  - **return** $\mathbf{d_x}$
- $\mathbf{acosl}_y \leftarrow$ **acos**$(\mathbf{d_y})$
- *# Checking whether the left bound is too large*
- *# to perform a reliable range reduction*
- *# That is, k_l would be off by more than one unit*
- **if** $\underline{\mathbf{d_x}} \notin [-2^{52}, 2^{52}]$:
  - $\mathbf{R}_{\text{left}} \leftarrow \mathbf{d_x}$
- **else**:
  - **if** $\underline{\mathbf{d_x}} < 0$:
    - $k_l \leftarrow \left\lfloor \mathrm{fl}_\triangledown\left(\frac{\underline{\mathbf{d_x}}}{\mathrm{fl}_\triangledown(\pi)}\right)\right\rfloor$
  - **elsif** $\underline{\mathbf{d_x}} > 0$:
    - $k_l \leftarrow \left\lfloor \mathrm{fl}_\triangledown\left(\frac{\underline{\mathbf{d_x}}}{\mathrm{fl}_\triangle(\pi)}\right)\right\rfloor$
  - **else**:
    - $k_l \leftarrow 0$
  - *# From here, the k_l computed is at most off by 1*
  - *# less than its exact value*
  - $\mathbf{R}_{\text{left}} \leftarrow$ **acos_k**$(k_l, \mathbf{acosl}_y) \cap \mathbf{d_x}$
  - **if** $\mathbf{R}_{\text{left}} = \varnothing$:
    - $\mathbf{R}_{\text{left}} \leftarrow$ **acos_k**$(k_l + 1, \mathbf{acosl}_y) \cap \mathbf{d_x}$
- *# Checking whether the right bound is too large*
- *# to perform a reliable range reduction*
- *# That is, k_r would be off by more than one unit*
- **if** $\overline{\mathbf{d_x}} \notin [-2^{52}, 2^{52}]$:
  - $\mathbf{R}_{\text{right}} \leftarrow \mathbf{d_x}$
- **else**:
  - **if** $\overline{\mathbf{d_x}} < 0$:
    - $k_r \leftarrow \left\lceil \mathrm{fl}_\triangle\left(\frac{\overline{\mathbf{d_x}}}{\mathrm{fl}_\triangle(\pi)}\right)\right\rceil$
  - **elsif** $\overline{\mathbf{d_x}} > 0$:
    - $k_r \leftarrow \left\lceil \mathrm{fl}_\triangle\left(\frac{\overline{\mathbf{d_x}}}{\mathrm{fl}_\triangledown(\pi)}\right)\right\rceil$
  - **else**:
    - $k_r \leftarrow 0$
  - *# From here, the k_r computed is at most off by 1*
  - *# more than its exact value*
  - **if** $k_r = k_l$:
    - $\mathbf{R}_{\text{right}} \leftarrow \mathbf{R}_{\text{left}}$
  - **else**:
    - $\mathbf{R}_{\text{right}} \leftarrow$ **acos_k**$(k_r, \mathbf{acosl}_y) \cap \mathbf{d_x}$
    - **if** $\mathbf{R}_{\text{right}} = \varnothing$:
      - $\mathbf{R}_{\text{right}} \leftarrow$ **acos_k**$(k_r - 1, \mathbf{acosl}_y) \cap \mathbf{d_x}$
- **return** $[\underline{\mathbf{R}_{\text{left}}}, \overline{\mathbf{R}_{\text{right}}}]$

---

function acos$(\mathbf{d_y} \in \mathbb{I})$:
- $\mathbf{I}'_\mathbf{y} \leftarrow \mathbf{d_y} \cap [-1, 1]$
- **if** $\mathbf{I}'_\mathbf{y} = \varnothing$:
  - **return** $\varnothing$
- **else**:
  - return [**acos_dn**$(\overline{\mathbf{I}'_\mathbf{y}})$, **acos_up**$(\underline{\mathbf{I}'_\mathbf{y}})$]

function acos_k$(k \in \mathbb{Z}, \mathbf{acosl}_y \in \mathbb{I})$:
*# Computes **acosl**$_y$ translated to the kth period*
- **if even**(k):
  - **return** $k\mathbf{\Pi} + \mathbf{acosl}_y$
- **else**:
  - **return** $(k+1)\mathbf{\Pi} - \mathbf{acosl}_y$

Interval Multivalued Inverse Functions

Relational Interval Arithmetic and its Use

Frédéric Goualard

`Frederic.Goualard@univ-nantes.fr`

Laboratoire d'Informatique de Nantes-Atlantique, UMR CNRS 6241

References

[Alefeld, 1968] Alefeld, G. (1968). *Intervallrechnung über den komplexen Zahlen und einie Anwendungen*. PhD thesis, University of Karlsruhe. Cited in [Hansen and Sengupta, 1981].

[Benhamou et al., 1999] Benhamou, F., Goualard, F., Granvilliers, L., and Puget, J.-F. (1999). Revising hull and box consistency. In *Proceedings of the sixteenth International Conference on Logic Programming (ICLP'99)*, pages 230–244, Las Cruces, USA. The MIT Press.

[Benhamou and Older, 1997] Benhamou, F. and Older, W. J. (1997). Applying interval arithmetic to real, integer and boolean constraints. *Journal of Logic Programming*, 32(1):1–24.

[Brönnimann et al., 2006] Brönnimann, H., Melquiond, G., and Pion, S. (2006). A proposal to add interval arithmetic to the C++ standard library. Technical Report N2137=06-0207, rev. 2, CIS Polytechnic Univ., École Normale Supérieure de Lyon, and INRIA.

[Ceberio and Granvilliers, 2000] Ceberio, M. and Granvilliers, L. (2000). Solving nonlinear systems by constraint inversion and interval arithmetic. In Campbell, J. A. and Roanes-Lozano, E., editors, *Proceedings of International Conference on Artificial Intelligence and Symbolic Computation*, volume 1930 of *Lecture Notes in Artificial Intelligence*, pages 127–141, Madrid, Spain. Springer-Verlag.

References continued

[Cleary, 1987] Cleary, J. G. (1987). Logical arithmetic. *Future Computing Systems*, 2(2):125–149.

[Ebers and Moll, 1954] Ebers, J. J. and Moll, J. L. (1954). Large-scale behaviour of junction transistors. *IRE Procs.*, 42:1761–1772.

[Granvilliers and Benhamou, 2001] Granvilliers, L. and Benhamou, F. (2001). Progress in the solving of a circuit design problem. *Journal of Global Optimization*, 20:155–168.

[Hansen and Walster, 2003] Hansen, E. and Walster, G. W. (2003). *Global Optimization using Interval Analysis*. Marcel Dekker, Inc.

[Hansen and Sengupta, 1981] Hansen, E. R. and Sengupta, S. (1981). Bounding solutions of systems of equations using interval analysis. *BIT*, 21:203–211.

[Hanson, 1968] Hanson, R. J. (1968). Interval arithmetic as a closed arithmetic system on a computer. Technical Report 197, Jet Propulsion Laboratory.

[Hickey et al., 1998] Hickey, T. J., Van Emden, M. H., and Wu, H. (1998). A unified framework for interval constraints and interval arithmetic. In Maher, M. and Puget, J.-F., editors, *Principles and Practice of Constraint Programming—CP98*, volume 1520 of *Lecture Notes in Computer Science*, pages 250–264. Springer-Verlag.

References continued

[Hyvönen and de Pascale, 1995] Hyvönen, E. and de Pascale, S. (1995). Inc++ library family for interval computations. In *Proceedings of the International Workshop on Applications of Interval Computations*.

[Kahan, 1968] Kahan, W. M. (1968). A more complete interval arithmetic. Technical report, University of toronto.

[Kearfott, 1991] Kearfott, R. B. (1991). Decomposition of arithmetic expressions to improve the behavior of interval iteration for nonlinear systems. *Computing*, 47:169–191.

[Mackworth, 1977] Mackworth, A. K. (1977). Consistency in networks of relations. *Artificial Intelligence*, 1(8):99–118.

[Moore, 1966] Moore, R. E. (1966). *Interval Analysis*. Prentice-Hall, Englewood Cliffs, N. J.

[Older, 1989] Older, W. J. (1989). Interval arithmetic specification. Research Report 89032, Computer Research Laboratory, Bell-Northern Research.

[Ratz, 1996] Ratz, D. (1996). Inclusion isotone extended interval arithmetic. Technical Report 5/1996, Institut für Angewandte Mathematik, Universität Karlsruhe.

References continued

[Vu et al., 2004] Vu, X.-H., Schichl, H., and Sam-Haroud, D. (2004). Using directed acyclic graphs to coordinate propagation and search for numerical constraint satisfaction problems. In *ICTAI '04: Proceedings of the 16th IEEE International Conference on Tools with Artificial Intelligence*, pages 72–81, Washington, DC, USA. IEEE Computer Society.

[Walster, 1998] Walster, G. W. (1998). The extended real interval system. Technical report.