

# The (near-)future IEEE 1788 standard for interval arithmetic

**Nathalie Revol**  
**INRIA - Université de Lyon**  
**LIP (UMR 5668 CNRS - ENS Lyon - INRIA - UCBL)**

SWIM 2015: 8th Small Workshop in Interval Methods  
Prague, 9-11 June 2015

# Agenda

Interval arithmetic: standardization

Standardization of interval arithmetic: IEEE P1788

Flavors

Overview of the IEEE-1788 standard

Intervals

Operations

Predicates

Exceptions and decorations

Conclusion

# Agenda

## Interval arithmetic: standardization

### Standardization of interval arithmetic: IEEE P1788

Flavors

Overview of the IEEE-1788 standard

Intervals

Operations

Predicates

Exceptions and decorations

## Conclusion

## Precious features

- ▶ **Fundamental theorem of interval arithmetic** (“**Thou shalt not lie**”): the returned result contains the sought result;
- ▶ **Brouwer theorem**: proof of existence (and uniqueness) of a solution;
- ▶ **ad hoc division**: gap between two semi-infinite intervals is preserved.

**Goal of a standardization**: keep the nice properties, have common definitions.

**Creation of the IEEE P1788 project**: Initiated by 15 attenders at Dagstuhl, Jan 2008. Project authorised as IEEE-WG-P1788, Jun 2008. Deadline: Oct 2015.

## Precious features

- ▶ **Fundamental theorem of interval arithmetic** (“**Thou shalt not lie**”): the returned result contains the sought result;
- ▶ **Brouwer theorem**: proof of existence (and uniqueness) of a solution;
- ▶ **ad hoc division**: gap between two semi-infinite intervals is preserved.

**Goal of a standardization:** keep the nice properties, have common definitions.

**Creation of the IEEE P1788 project:** Initiated by 15 attenders at Dagstuhl, Jan 2008. Project authorised as IEEE-WG-P1788, Jun 2008. Deadline: Oct 2015.

## Precious features

- ▶ **Fundamental theorem of interval arithmetic** (“**Thou shalt not lie**”): the returned result contains the sought result;
- ▶ **Brouwer theorem**: proof of existence (and uniqueness) of a solution;
- ▶ **ad hoc division**: gap between two semi-infinite intervals is preserved.

**Goal of a standardization**: keep the nice properties, have common definitions.

**Creation of the IEEE P1788 project**: Initiated by 15 attenders at Dagstuhl, Jan 2008. Project authorised as IEEE-WG-P1788, Jun 2008. Deadline: Oct 2015.

# Agenda

Interval arithmetic: standardization

Standardization of interval arithmetic: IEEE P1788

Flavors

Overview of the IEEE-1788 standard

Intervals

Operations

Predicates

Exceptions and decorations

Conclusion

## Flavors: Moore's classic IA

Only non-empty and bounded intervals.

Adopted as the common basis for all flavors,  
but limited because useful intervals are missing.



## Flavors: Moore's classic IA

Only non-empty and bounded intervals.

Adopted as the common basis for all flavors,  
but limited because useful intervals are missing.

## Flavors: set-based IA

Also allows unbounded intervals and empty set.

Very famous theory, sound.

Adopted: the only flavor currently defined in the standard.

## Flavors: set-based IA

Also allows unbounded intervals and empty set.

Very famous theory, sound.

Adopted: the only flavor currently defined in the standard.

## Flavors: set-based IA

Also allows unbounded intervals and empty set.

Very famous theory, sound.

Adopted: the only flavor currently defined in the standard.

## Flavors: Kaucher, modal

$[1, 2]$  and  $[2, 1]$  are allowed.

Unbounded intervals are not permitted.

Hook provided, but theory still missing.

Hopefully, it will be added in the revision of the standard, in 2025!

## Flavors: Kaucher, modal

$[1, 2]$  and  $[2, 1]$  are allowed.  
Unbounded intervals are not permitted.

Hook provided, but theory still missing.

Hopefully, it will be added in the revision of the standard, in 2025!

## Flavors: Kaucher, modal

$[1, 2]$  and  $[2, 1]$  are allowed.

Unbounded intervals are not permitted.

Hook provided, but theory still missing.

Hopefully, it will be added in the revision of the standard, in 2025!

## Flavors: cset...

Cset:  $1/[0, 1] = \mathbb{R}$ .

Conservative:  $\sqrt{[-2, 1]} = \text{NaN}$ ?

Discussed, alluded to, apparently cset is being developed... concurrently.



## Flavors: cset...

Cset:  $1/[0, 1] = \mathbb{R}$ .

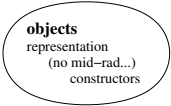
Conservative:  $\sqrt{[-2, 1]} = \text{Nal?}$

Discussed, alluded to, apparently cset is being developed... concurrently.


## IEEE-1788 standard: the big picture

<b>LEVEL1</b> mathematics	
<b>LEVEL2</b> implementation or discretization	
<b>LEVEL3</b> computer representation	
<b>LEVEL4</b> bits	

## IEEE-1788 standard: the big picture

<b>LEVEL1</b> mathematics	
<b>LEVEL2</b> implementation or discretization	
<b>LEVEL3</b> computer representation	
<b>LEVEL4</b> bits	

## IEEE-1788 standard: the big picture

<b>LEVEL1</b> mathematics	 <p><b>objects</b> representation (no mid-rad...) constructors</p> <p><b>operations</b> arithmetic set interval</p>
<b>LEVEL2</b> implementation or discretization	
<b>LEVEL3</b> computer representation	
<b>LEVEL4</b> bits	

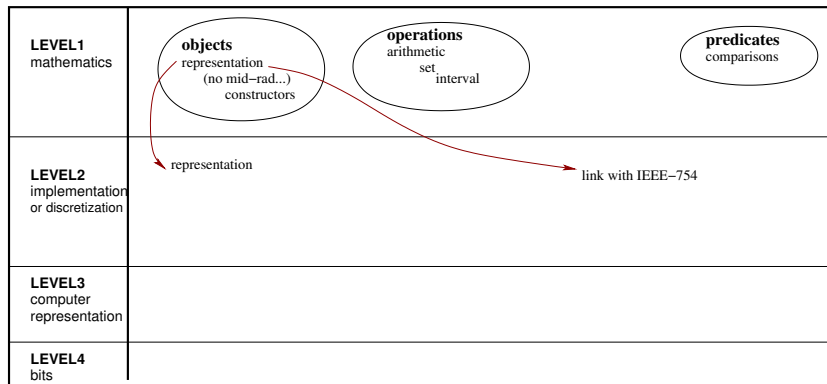
## IEEE-1788 standard: the big picture

<b>LEVEL1</b> mathematics	<b>objects</b> representation (no mid-rad...) constructors	<b>operations</b> arithmetic set interval	<b>predicates</b> comparisons
<b>LEVEL2</b> implementation or discretization			
<b>LEVEL3</b> computer representation			
<b>LEVEL4</b> bits			

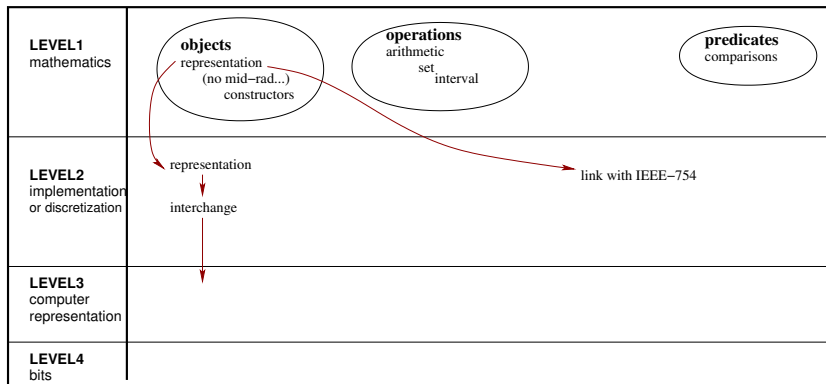
## IEEE-1788 standard: the big picture

<b>LEVEL1</b> mathematics	<p><b>objects</b> representation (no mid-rad...) constructors</p> <p><b>operations</b> arithmetic set interval</p> <p><b>predicates</b> comparisons</p>
<b>LEVEL2</b> implementation or discretization	representation
<b>LEVEL3</b> computer representation	
<b>LEVEL4</b> bits	

## IEEE-1788 standard: the big picture

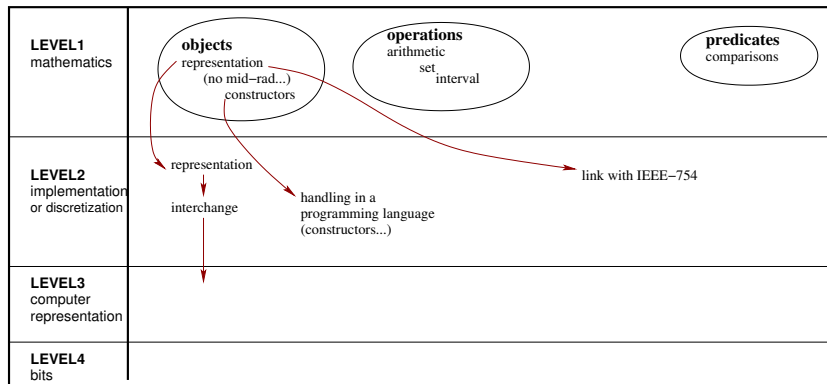


## IEEE-1788 standard: the big picture

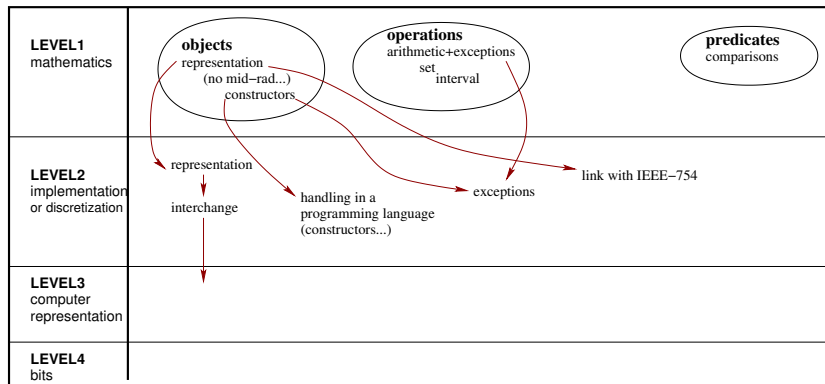




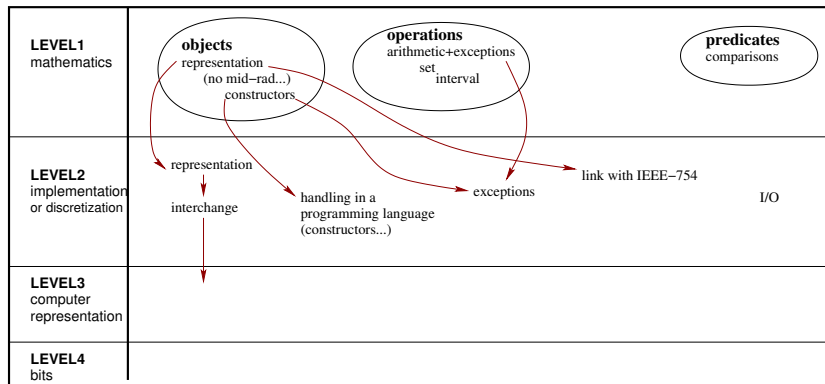
## IEEE-1788 standard: the big picture



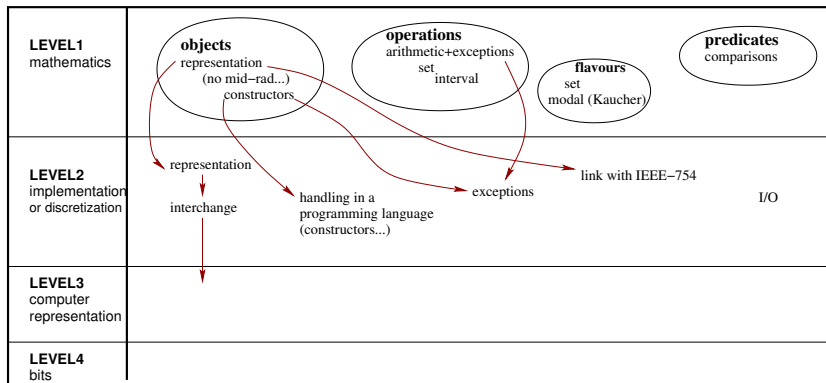
## IEEE-1788 standard: the big picture



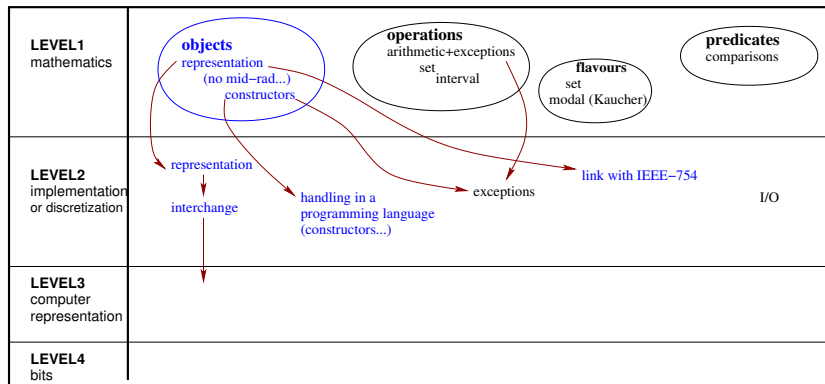
## IEEE-1788 standard: the big picture



# IEEE-1788 standard: the big picture



## IEEE-1788 standard: the big picture

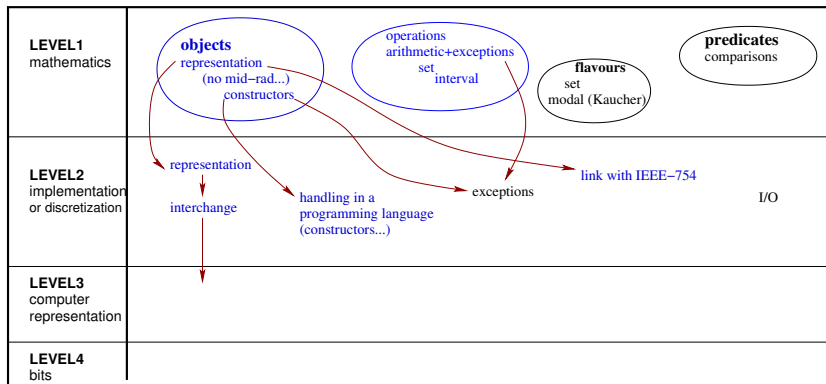


## IEEE-1788 standard: intervals

**Representation** (some intervals, like  $\emptyset$  in the set-based flavor, may need a special representation):

- ▶ **adopted representation: by the bounds (inf-sup);**
- ▶ uncertain form: value  $\pm$  ulp-count and optional direction.

# IEEE-1788 standard: the big picture



## IEEE-1788 standard: arithmetic operations

At level 1:

- ▶  $+$ ,  $-$ ,  $\times$ : everybody agrees
- ▶  $/$ :  $[2, 3]/[-1, 2]$ : 2 semi-infinities,  $\mathbb{R}$ ?  
 $\mathbb{R}$  for closedness (in set-based flavor)
- ▶  $\sqrt{[-1, 2]}$ ?  $[0, \sqrt{2}]$ .

no NaN but exceptional condition.

At level 2:

no NaN but exceptional condition.



## IEEE-1788 standard: arithmetic operations

At level 1:

- ▶  $+$ ,  $-$ ,  $\times$ : everybody agrees
- ▶  $/$ :  $[2, 3]/[-1, 2]$ : 2 semi-infinities,  $\mathbb{R}$ ?  
 $\mathbb{R}$  for closedness (in set-based flavor)
- ▶  $\sqrt{[-1, 2]}$ ?  $[0, \sqrt{2}]$ .

no NaN but exceptional condition.

At level 2:

no NaN but exceptional condition.

## IEEE-1788 standard: arithmetic operations

At level 1:

- ▶  $+$ ,  $-$ ,  $\times$ : everybody agrees
- ▶  $/$ :  $[2, 3]/[-1, 2]$ : 2 semi-infinities,  $\mathbb{R}$ ?  
 $\mathbb{R}$  for closedness (in set-based flavor)
- ▶  $\sqrt{[-1, 2]}$ ?  $[0, \sqrt{2}]$ .

no NaN but exceptional condition.

At level 2:

no NaN but exceptional condition.

## IEEE-1788 standard: arithmetic operations

At level 1:

- ▶  $+$ ,  $-$ ,  $\times$ : everybody agrees
- ▶  $/$ :  $[2, 3]/[-1, 2]$ : 2 semi-infinities,  $\mathbb{R}$ ?  
 $\mathbb{R}$  for closedness (in set-based flavor)
- ▶  $\sqrt{[-1, 2]}$ ?  $[0, \sqrt{2}]$ .

no NaN but exceptional condition.

At level 2:

no NaN but exceptional condition.

## IEEE-1788 standard: arithmetic operations

At level 1:

- ▶  $+$ ,  $-$ ,  $\times$ : everybody agrees
- ▶  $/$ :  $[2, 3]/[-1, 2]$ : 2 semi-infinities,  $\mathbb{R}$ ?  
 $\mathbb{R}$  for closedness (in set-based flavor)
- ▶  $\sqrt{[-1, 2]}$ ?  $[0, \sqrt{2}]$ .

no NaN but exceptional condition.

At level 2:

no NaN but exceptional condition.

# IEEE-1788 standard: operations on intervals

## Example: midpoint

What is the midpoint of

- ▶  $\mathbb{R}$ ?     0?
- ▶  $[2, +\infty)$ ?     `MaxReal` in  $\mathbb{F}$ ? (at Level 2)
- ▶  $\emptyset$ ?     ????: undefined at Level 1, NaN at Level 2.

# IEEE-1788 standard: operations on intervals

## Example: midpoint

What is the midpoint of

- ▶  $\mathbb{R}$ ?     0?
- ▶  $[2, +\infty)$ ?     MaxReal in  $\mathbb{F}$ ? (at Level 2)
- ▶  $\emptyset$ ?     ????: undefined at Level 1, NaN at Level 2.

# IEEE-1788 standard: operations on intervals

## Example: midpoint

What is the midpoint of

- ▶  $\mathbb{R}$ ?    0?
- ▶  $[2, +\infty)$ ?    `MaxReal` in  $\mathbb{F}$ ? (at Level 2)
- ▶  $\emptyset$ ?    ????: undefined at Level 1, NaN at Level 2.

# IEEE-1788 standard: operations on intervals

## Example: midpoint

What is the midpoint of

- ▶  $\mathbb{R}$ ?    0?
- ▶  $[2, +\infty)$ ?    `MaxReal` in  $\mathbb{F}$ ? (at Level 2)
- ▶  $\emptyset$ ?    ????: undefined at Level 1, NaN at Level 2.



# IEEE-1788 standard: operations on intervals

## Example: midpoint

What is the midpoint of

- ▶  $\mathbb{R}$ ?     0?
- ▶  $[2, +\infty)$ ?     MaxReal in  $\mathbb{F}$ ? (at Level 2)
- ▶  $\emptyset$ ?     ????: undefined at Level 1, NaN at Level 2.

# IEEE-1788 standard: operations on intervals

## Example: midpoint

What is the midpoint of

- ▶  $\mathbb{R}$ ?    0?
- ▶  $[2, +\infty)$ ?    MaxReal in  $\mathbb{F}$ ? (at Level 2)
- ▶  $\emptyset$ ?    ????: undefined at Level 1, NaN at Level 2.

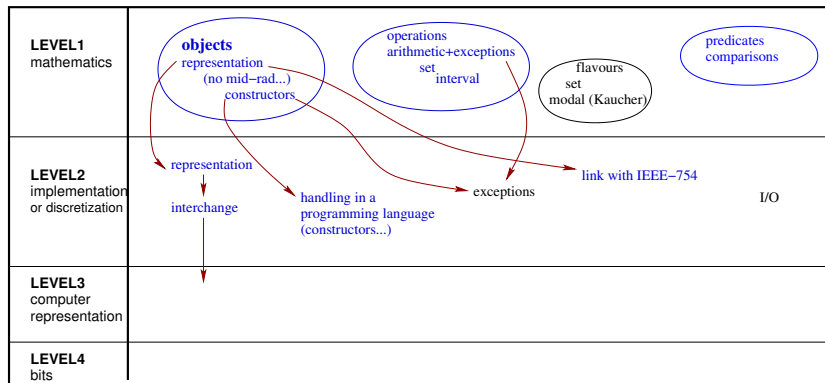
# IEEE-1788 standard: operations on intervals

## Example: midpoint

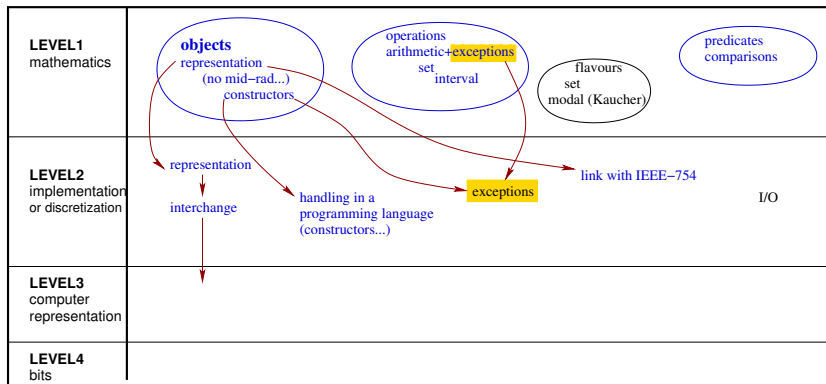
What is the midpoint of

- ▶  $\mathbb{R}$ ?    0?
- ▶  $[2, +\infty)$ ?    MaxReal in  $\mathbb{F}$ ? (at Level 2)
- ▶  $\emptyset$ ?    ???: undefined at Level 1, NaN at Level 2.

# IEEE-1788 standard: the big picture



# IEEE-1788 standard: the big picture



## IEEE-1788: exception raised by a constructor

On the mathematical model & implementation sides:

`nums2interval(2,1)`, i.e.  $\{x \in \mathbb{R} \mid 2 \leq x \leq 1\}$ ?

Empty? Exception (error)? Notion of Not-an-Interval ("NaI")?

The revenge of the NaI! (in set-based flavor).

Meaningful in Kaucher arithmetic: let this possibility open.

## IEEE-1788: exception raised by a constructor

On the mathematical model & implementation sides:  
`nums2interval(2,1)`, i.e.  $\{x \in \mathbb{R} \mid 2 \leq x \leq 1\}$ ?

Empty? Exception (error)? Notion of Not-an-Interval (“NaI”)?  
The revenge of the NaI! (in set-based flavor).  
Meaningful in Kaucher arithmetic: let this possibility open.

## IEEE-1788: exception raised by a constructor

On the mathematical model & implementation sides:

`nums2interval(2,1)`, i.e.  $\{x \in \mathbb{R} \mid 2 \leq x \leq 1\}$ ?

Empty? Exception (error)? Notion of Not-an-Interval (“Nal”)?

The revenge of the Nal! (in set-based flavor).

Meaningful in Kaucher arithmetic: let this possibility open.



## IEEE-1788: exception raised by a constructor

On the mathematical model & implementation sides:

`nums2interval(2,1)`, i.e.  $\{x \in \mathbb{R} \mid 2 \leq x \leq 1\}$ ?

Empty? Exception (error)? Notion of Not-an-Interval (“Nal”)?

The revenge of the Nal! (in set-based flavor).

Meaningful in Kaucher arithmetic: let this possibility open.

## IEEE-1788 standard: exception handling

**Exceptions must be handled** in some way, even if exceptions do look... exceptional. (It must have been the same for exception handling in IEEE-754 floating-point arithmetic.)

**Best way to handle exceptions?** To avoid global flags, “tags” attached to each interval: decorations.

Decorated intervals 

**Discussions** about what should be in the decorations.

For common intervals: `com` for common.

For set-based flavor: `com` for common, `dac` for defined and continuous, `def` for defined, `trv` for no information, `ill` for nowhere defined.

## IEEE-1788: arguments outside the domain

How should  $f(x)$  be handled when  $x$  is not included in the domain of  $f$ ? E.g.  $\sqrt{[-1, 2]}$ ?

- ▶ exit?
- ▶ return Nal (Not an Interval)? I.e. handle exceptional values such as Nal and infinities?
- ▶ return the set of every possible limits  $\lim_{y \rightarrow x} f(y)$  for every possible  $x$  in the domain of  $f$  (but not necessarily  $y$ )?
- ▶ intersect  $x$  with the domain of  $f$  prior to the computation, silently?
- ▶ intersect  $x$  with the domain of  $f$  prior to the computation and mention it

## Remark: arguments outside the domain

(Rump, Dagstuhl seminar 09471, 2009)

$$\begin{aligned} f(x) &= |x - 1| \\ g(x) &= (\sqrt{x - 1})^2 \end{aligned} \quad \text{I know, it is not the best way of writing it. . .}$$

What happens if  $x = [0, 1]$ ?

$$\begin{aligned} f(x) &= [0, 1] \\ g(x) &= [0] \end{aligned}$$

Without exception handling, **the Thou shalt not lie principle is not valid.**

One has to check whether there has been a *possibly undefined* operation. . . Unexperienced programmers will not do it.

In other words, it is not possible to protect people from getting wrong results, however hard we try.

## Remark: arguments outside the domain

(Rump, Dagstuhl seminar 09471, 2009)

$$\begin{aligned} f(x) &= |x - 1| \\ g(x) &= (\sqrt{x - 1})^2 \end{aligned} \quad \text{I know, it is not the best way of writing it. . .}$$

What happens if  $x = [0, 1]$ ?

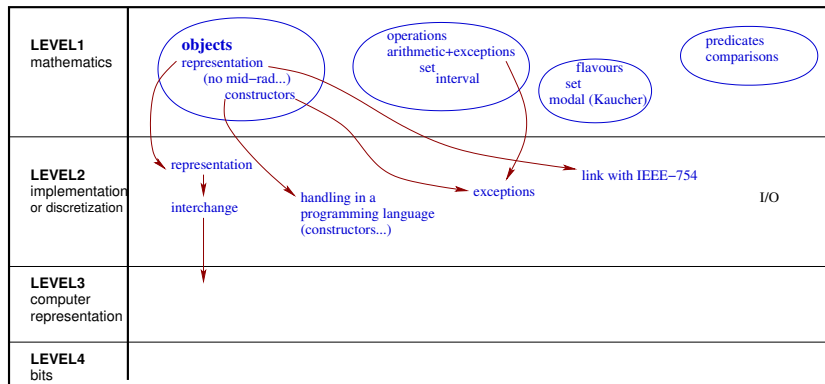
$$\begin{aligned} f(x) &= [0, 1] \\ g(x) &= [0] \end{aligned}$$

Without exception handling, **the Thou shalt not lie principle is not valid.**

One has to check whether there has been a *possibly undefined* operation. . . Unexperienced programmers will not do it.

In other words, it is not possible to protect people from getting wrong results, however hard we try.

# IEEE-1788 standard: the big picture



## Another hot topic: EDP

EDP means Exact Dot Product.

EDP = computing exactly the dot product of two floating-point vectors.

Either in hardware or in software.

Recommended.

## Another hot topic: EDP

EDP means Exact Dot Product.

EDP = computing exactly the dot product of two floating-point vectors.

Either in hardware or in software.

Recommended.



# Agenda

Interval arithmetic: standardization

Standardization of interval arithmetic: IEEE P1788

Flavors

Overview of the IEEE-1788 standard

Intervals

Operations

Predicates

Exceptions and decorations

Conclusion

## Existing reference implementations

**libieee1788** by Marco Nehmeier,  
set-based flavor, in C++, based on MPFR.

**interval 0.1.5** by Oliver Heimlich,  
in GNU Octave.

The title of the talk was

## The (near-)future IEEE 1788 standard for interval arithmetic

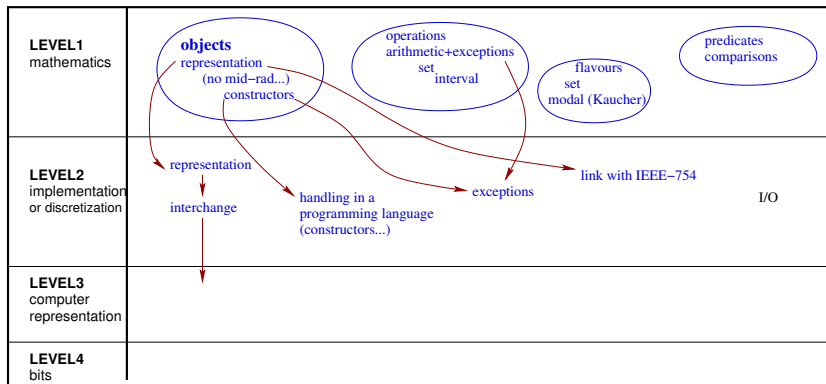
### The news are:

- ▶ the final draft of the standard has gone through the final approval stage June 5 (last Friday);
- ▶ the standard should be published July 11.

### Future work:

- ▶ IEEE-1788.1 "Standard for Interval Arithmetic (Simplified)" has been approved January 30: Ned Nedialkov is in charge
- ▶ IEEE 754-2008 "Standard for Binary Floating-Point Arithmetic" (also includes decimal arithmetic) expires in 2018: it is time to start the revision process.

## IEEE-1788 standard: the big picture



## IEEE-1788 standard: the original project

**P1788 Scope:** “This standard specifies basic [interval arithmetic](#) (IA) operations selecting and following one of the commonly used [mathematical interval models](#). This standard supports the IEEE-754/2008 floating point types of practical use in interval computations. [Exception conditions](#) will be defined and standard handling of these conditions will be specified. Consistency with the model is tempered with practical considerations based on input from representatives of vendors and owners of existing systems. The standard provides a layer between the hardware and the programming language levels. It [does not mandate](#) that any operations be implemented in hardware. It [does not define any realization](#) of the basic operations as functions in a programming language.”