IEEE 1788 Working Group for the Standardization of Interval Arithmetic
a brief overview

Nathalie Revol
INRIA – LIP - ENS de Lyon – France

TMW VII - 14-17 December 2011 - Key West
Agenda

Interval Arithmetic
  Important facts that underlie many discussions and decisions

Standardization of interval arithmetic: IEEE P1788
  Facts about the working group
  Motions and topics of discussion
  Motions adopted
  Motions not adopted
  Personal view
  Exception handling

Conclusion

More on decorations
Agenda

Interval Arithmetic

Important facts that underlie many discussions and decisions

Standardization of interval arithmetic: IEEE P1788

Facts about the working group

Motions and topics of discussion

Motions adopted

Motions not adopted

Personal view

Exception handling

Conclusion

More on decorations
Pros and cons of interval arithmetic

**Pros:**

- **“thou shalt not lie”:** guarantee that the result belongs to the resulting interval;
- **computing in the large:** computing with whole set, global optimization;
- **Brouwer theorem made effective:** if $f(K) \subseteq K$ then $f$ has a fixed point in $K$. As this can be checked, existence and uniqueness can be proven.

**Cons:**

- **implementation** requires a specific algorithm, not only changing float into interval;
- **overestimation** that can make a computed interval (much) wider than the exact range of the mathematical function on the same input interval.
Agenda

Interval Arithmetic

Important facts that underlie many discussions and decisions

Standardization of interval arithmetic: IEEE P1788

- Facts about the working group
- Motions and topics of discussion
- Motions adopted
- Motions not adopted
- Personal view
- Exception handling

Conclusion

More on decorations
The P1788 Project

Belief: interval arithmetic is mature enough to undergo a common definition.

Goal: standardize interval arithmetic.

How P1788’s work is done

- The bulk of work is carried out by email, with electronic voting.
- Motions are proposed, seconded; three weeks discussion period; three weeks voting period.
- IEEE has given us a four year deadline...which expires soon, we will ask for a 2-years extension.
- One “in person” meeting per year is planned—last one was be July 25th, 2011, in Tübingen, Germany, during the Arith 20 conference.
- IEEE auspices: 1 report + 1 teleconference quarterly
IEEE-1788 WG: some facts

Since October 2008: very active mailing list
over 150 participants, over 20 nationalities, over 4400 messages

Work already done:
adoption of officers, of procedures and policy
roster of (voting or not) members: 88 members, 18 nationalities
31 motions handled.

URL of the working group:
http://grouper.ieee.org/groups/1788/
or google 1788 interval arithmetic.
Motions discussed so far

1. Provisional standard notation for intervals
2-14. Levels structure for standardisation process
3. Standard is based on $\mathbb{R}$ not $\mathbb{R}^*$
4-24. Restrict standard to 754 systems, rounded operations
5-10. Arithmetic operations and elementary functions
6. Multi- & mixed-format interval support
9. Exact dot product
11-12. Reverse Arithmetic Operations
13-14-20. Comparison Relations
21. Overlapping intervals
16-19-23. inf/sup and mid/rad
17. IO
28. Containment only

Nathalie Revol  INRIA – LIP – ENS de Lyon – France

IEEE 1788 for Interval Arithmetic
Motions adopted so far

1. Provisional standard notation for intervals
2-14. Levels structure for standardisation process
   3. Standard is based on $\mathbb{R}$ not $\mathbb{R}^*$
24. Restrict standard to 754 systems, rounded operations
5-10. Arithmetic operations and elementary functions
   6. Multi- & mixed-format interval support
   9. Exact dot product
12. Reverse Arithmetic Operations
13-14-20. Comparison Relations
21. Overlapping intervals
16-19. inf/sup and mid/rad
17. IO
8-18. Exception handling: decorations
"Quarrels of chapels" - Parochial quarrels
My dream
My feelings last summer
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, flags attached to each interval: decorations.

Decorated intervals

Discussions about what should be in the decorations (defined and continuous, defined, no-information, nowhere defined).
Exception: 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 NaI (Not an Interval)? I.e. handle exceptional values such as NaI and infinities?
- return the set of every possible limits $\lim_{y \to x} f(y)$ 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
## Agenda

**Interval Arithmetic**
- Important facts that underlie many discussions and decisions

**Standardization of interval arithmetic: IEEE P1788**
- Facts about the working group
- Motions and topics of discussion
- Motions adopted
- Motions not adopted
- Personal view
- Exception handling

## Conclusion

**More on decorations**
To conclude

The IEEE committee will have to

- complete the list of things that have to be part (or not) of the standard, and how they are part of it
  and you can help us!
- discuss every point, its pro and cons (using counterexamples) and you can help us!
- agree on the most sensible choice...
  and then you will vote to tell us if we were right!

See you in 2 (or 4, or 6) years time, to introduce you the new standard!
To conclude

The IEEE committee will have to

- complete the list of things that have to be part (or not) of the standard, and how they are part of it
  and you can help us!

- discuss every point, its pro and cons (using counterexamples)
  and you can help us!

- agree on the most sensible choice...
  and then you will vote to tell us if we were right!

See you in 2 (or 4, or 6) years time, to introduce you the new standard!
To conclude

The IEEE committee will have to

- complete the list of things that have to be part (or not) of the standard, and how they are part of it
  and you can help us!

- discuss every point, its pro and cons (using counterexamples)
  and you can help us!

- agree on the most sensible choice…
  and then you will vote to tell us if we were right!

See you in 2 (or 4, or 6) years time, to introduce you the new standard!
To conclude

The IEEE committee will have to

- complete the list of things that have to be part (or not) of the standard, and how they are part of it
  and you can help us!

- discuss every point, its pro and cons (using counterexamples)
  and you can help us!

- agree on the most sensible choice . . .
  and then you will vote to tell us if we were right!

See you in 2 (or 4, or 6) years time, to introduce you the new standard!
Agenda

Interval Arithmetic
  Important facts that underlie many discussions and decisions

Standardization of interval arithmetic: IEEE P1788
  Facts about the working group
  Motions and topics of discussion
  Motions adopted
  Motions not adopted
  Personal view
  Exception handling

Conclusion

More on decorations
Exception: 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 NaI (Not an Interval)? I.e. handle exceptional values such as NaI and infinities?
- return the set of every possible limits $\lim_{y \to x} f(y)$ 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
Motions 7, 8, 15 and 18: Exceptions

**Issue:** How to handle exceptions efficiently.

- Typical examples:
  - (a) Invalid interval constructor
    
    ```
    interval(3,2)     interval("[2.4,3;5]"")
    ```
    —interface between interval world and numbers or text strings.
  - (b) Elementary function evaluated partly or wholly outside domain
    
    ```
    sqrt([-1,4])     log([-4,-1])     [1,2]/[0,0]
    ```

- Type (a) can simply cause nonsense if ignored.
- Type (b) are crucial for applications that depend on fixed-point theorems; but can be ignored by others, e.g. some optimisation algorithms.
Motions 7 and 8: Exceptions, cont.

What to do? A complicated issue.

- Risk that (Level 3) code to handle exceptions will slow down interval applications that don’t need it.
- One approach to type (a) is to define an NaI ”Not an Interval” datum at level 2, encoded at level 3 within the two FP numbers that represent an interval.
Motion 8: Exceptions by Decorations

- Alternative (Motion 8): An extra tag or decoration field (1 byte?) in level 3 representation.
- Divided into subfields that record different kinds of exceptional behaviour.
- Decoration is optional, can be added and dropped.
  - To compute at full speed, use “bare” intervals and corresponding “bare” elementary function library.
  - “Decorated” library records exceptions separate from numbers, hence code has fewer IFs & runs fast too. (We hope!)
Motions 8, 15 and 18: Decoration issues

Decorations look promising but many Qs exist:

- Bare (double) interval is 16-byte object. Decoration increases this. Can compilers efficiently allocate memory for large arrays of such objects?
- Some proposed decoration-subfields record events in the past; others are properties of the current interval. Can semantic inconsistencies arise?
- Can decoration semantics be specified at Level 2 . . .
- . . . such that correctness of code can be proven . . .
- . . . and K.I.S.S. is preserved?

Much work on exceptions remains: list, order . . .
Remark: arguments outside the domain

Problematic example (Rump, Dagstuhl seminar 09471, 2009).

\[ f(x) = |x - 1| \]
\[ g(x) = (\sqrt{x - 1})^2 \]

I know, it is not the best way of writing it...

What happens if \( x = [0, 1] \)?

With the adopted definitions of operations,

\[ f(x) = [0, 1] \]
\[ g(x) = [0] \]

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.
Remark: arguments outside the domain

Problematic example (Rump, Dagstuhl seminar 09471, 2009).

$$f(x) = |x - 1|$$
$$g(x) = (\sqrt{x - 1})^2$$

I know, it is not the best way of writing it... What happens if $$x = [0, 1]$$?

With the adopted definitions of operations,

$$f(x) = [0, 1]$$
$$g(x) = [0]$$

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.