Position: That the Standard for Computing with Intervals Have Only One Level 1 Requirement: Containment.

G. William Walster

August 1, 2011

Acknowledgement 1 *Thanks to R. Baker Kearfott for his thoughtful comments and suggestions.*

Because no accuracy information is retained in finite-precision normalized floating-point numbers, it is impossible to create a standard for computing with them that is accuracy-based. Rather, any floating-point standard must specify how floating-point computing is implemented.

Intervals are different because they include their accuracy. By including in an interval implementation standard only the most fundamental interval accuracy requirement, containment, it is possible to create an interval standard that is free of any other implementation requirements.

A containment-only interval standard is desirable because any other implementation requirements must necessarily preclude "quality of implementation features" that improve interval computing. For example if an implementation-specific interval standard does not include extended values, they may not exist in a standard-compliant implementation. Conversely, there must be no requirement to implement extended intervals in a containment-only standard. Extended intervals are an optional "quality of implementation feature" in a containment-only standard.

A containment-only interval standard must leave level 2 through 4 implementation details completely unspecified. The one and only level 1 implementation requirement must be to enclose the smallest set of values (the containment set) that must be contained in the interval result of any evaluated arithmetic expression or function defined therefrom. Given this one and only requirement, optional "quality of implementation features" include, but are not limited to: speed; narrow width; ease of use; and both convenient and transparent language syntax and semantics.

A containment-only interval standard must leave internal binary representations opaque with no requirement for users to be able to manipulate bits other than through supplied containment-safe library routines. There must be no requirement for binary compatibility between hardware platforms, or even

different vendor's compilers. How any vendor internally represents intervals on their hardware is also a "quality of implementation feature". Similarly, there must be no requirement for a particular set of low level interval operations. The ultimate "proof of the pudding" for computing with intervals is implementation quality in terms of speed, narrow width, and ease of interval algorithm development and use.

With a containment-only interval standard, there are no constrains on how to achieve quality. For example, a compiler might (as a "quality of implementation feature") automate the trade-off between speed and narrow width, depending on what is good for a given algorithm. What is good for a given algorithm might be speed given a narrow width requirement, or it might be width given a time limit.

A containment-only interval standard must define containment sets for any evaluated arithmetic expression or function defined therefrom. Otherwise, it is impossible to validate any given implementation; to know how much "quality of implementation" (in the form of interval width) remains available to be achieved; or to determine that any given implementation (no mater the standard on which it is based) produces a containment failure. Defining containment sets for all evaluated numerical expressions and functions defined therefrom is a purely mathematical problem that does not depend on how computing with intervals is implemented. This, and not level 2 through 4 implementation specifications, is the sole content of a containment-only interval computing standard.