

Quantifying Compromises in Correctness

Eerke Boiten, University of Kent

University of Bristol, 16 November 2006

credits: Dan Grundy, John Derrick

Realism vs Idealism

Software *isn't* correct

Why not?

Can it be? Should it be? How do we accommodate this?

Summary of this talk

- Go with the realists
- ... but quantify the correctness
- ... i.e., *measure* it
- ... and don't lose the correctness ambition: where is it, can we get closer?

Contribution: common understanding;
increased applicability of FM.

Abstraction vs. Completeness

- Claim: in formal methods, they're natural enemies.
- Specify a garbage collector: how often, how many cells?
- \mathbb{Z} vs. $\mathbb{Z}_{2^{\text{maxint}}}$
- real number vs. float
- unbounded vs. finite buffer

Key idea

- Not directly from $\{0,1\}$ to $[0..1]$
(Meaning? How to improve?)
- Indexing by an unbounded measure
- Infinity represents the ideal
- Correctness monotonic (or better!)

Quantitative Correctness Specification

Given language L (with \equiv).

QCS = (M, S, d) where

M is a measurement space (often: \mathbb{N})

$S: M \rightarrow L$ is an indexed specification

d is a metric on L

Notion of limit

Interpretation of QCS

(M, S, d) is considered equivalent to the limit of S under d

(meaningless if it hasn't)

S is an indexed set of compromises

(Alternative approach: ordering on L , lub of S .)

Example 1: Resource Bounds in Formal Specs

$$int_n =_{def} -n \dots n-1$$

$$incr_n =_{def}$$

$$c : [c \in int_n ,$$

$$c \in int_n \wedge c \bmod 2n = (c_0 + 1) \bmod 2n]$$

Index set: \mathbb{N}^+ , metric: execute until results differ, p times, distance is 2^{-p} (geo)

Example 2: Model Checking

- Telephony feature interaction scenario: collection of features specified by F
- Check no interactions with n phones
- Establishes
 $phones \leq n \Rightarrow F$

Example 3: Numerical Algorithms

- Specifications are mathematical
- Algorithms come with precision guarantees, normally dependent on representation size used and/or time
- Can increase precision arbitrarily in principle (at expense of resource)

Example 4: Testing

- Harder to find examples: most measurements don't guarantee correctness if maximised.
- W-machine, X-method: state machines, chains of n duplicated states of which last erroneous

Example 5: Provable Security in Cryptography

- Perfect security is the ideal
- Computationally secure schemes approximate that arbitrarily for increased parameter (key size)
- Extra requirement: *fast* convergence to the limit

So what is it good for?

- Framework for understanding commonality in practical software engineering
- Increases scope for formal specification and development (and thus ...)

Back to Formal Methods: QCS development

- Abstract specifications to start with
- Introduce QCS only when reference to measurement (resource bound) becomes necessary
- Refine QCS uniformly element-wise if possible (preserving limit, measure is “universal parameter”)

... with Approximations

- Compromise: fix value of n
(size of buffer, value of maxint, ...)
- Not "correct" so ...
- ... postpone until as late as possible
(profiling, targeting to architecture)
- *but* which properties *are* preserved?

Metrics on specifications

- Program length
(e.g. buffer of size n distinguished from unbounded one after ...)
- Input/program distribution
(distance is prob density of all inputs/programs that have "different" outcomes)