Summaries of Selected Dissertations

Murali Sitaraman

RESOLVE Software Research Group (RSRG)
Clemson University
http://www.cs.clemson.edu/group/resolve

We gratefully acknowledge NSF grants CCF-0811748, CCF-1161916, and DUE-1022941.
RESOLVE Language Characteristics

• Support for both **formal specifications** and **implementation code** (separable interfaces)
  – **Formal specifications**: Syntactic slots for pre/post conditions, loop invariants, etc.
  – Separating specs and code for component reuse

• Support for **Clean Semantics**
  – Must be able to account for (and restrict) harmful referencing, aliasing, and mutation
  – Permit explicit pointer usage where unavoidable
Heather Harton Dissertation

• Mechanical and Modular Verification Condition Generation For Object-Based Software, 2011.
Verifying Compiler beneath the IDE

• The RESOLVE verifying compiler supports IDE actions
• The compiler is capable of automatically verifying correctness of *practical* programs
• Aimed at addressing Tony Hoare’s seminal verifying compiler ‘grand challenge’
  – A future where no software is considered complete until formally verified
Verifying Code Correctness

• VCs (Verification Conditions) are necessary and sufficient conditions to prove implementation correctness

• VCs arise from multiple sources:
  – Ensures clause of an operation’s implementation under verification
  – Requires clause of a called operation
  – Establishing correctness of programmer-supplied assertions (e.g., loop and representation invariants)
  – Proof of termination
Hampton Smith Dissertation

RESOLVE Language Characteristics

• **Mathematical theories** formally define various notions, notations, and results

• RESOLVE allows for creation and use of new theories
  – Unlikely that *all* software can be adequately explained with only a handful of mathematical models
  – Though always continue to encourage reuse of existing ones when possible

• Prover can use math results to prove VCs
Minimalist Prover

• If software specifications and code are properly designed and the underlying mathematical theories are properly engineered and include suitable results, then a simple prover is sufficient to prove VCs.

• A prototyping of the prover

• Sound, but necessarily incomplete
Svetlana Drachova Dissertation

• Teaching and Assessment of Mathematical Principles for Software Correctness Using a Reasoning Concept Inventory, 2013.
Teaching Reasoning

• Institutionalization of specification and reasoning principles in two required course for CS majors

• Development of an inventory of reasoning principles and specific learning objectives

• Multi-year assessment, refinement, and improvement