The Final Step in the Evolution of Programming

Aaron Stump Cesare Tinelli

Computational Logic Center
Computer Science
The University of Iowa

Grad Orientation '12

The Evolution of Programming

Machine code, Assembly language.

Procedural abstraction, Structured programming.

Fortran, LISP, Cobol, Algol (1950s-1960s)
C (early 1970s)

Object-orientation

Simula, Smalltalk (1960s-1970s)

Typed Functional Programming

ML (early 1970s)

Logic Programming

Prolog (early 1970s)

Tweaking

C++, Scheme, Java, Standard ML (1980s)

More Tweaking

Haskell, OCaml, Scala, Python (1990s-2000s)

If We're Just Tweaking...

If We're Just Tweaking...

We must be done!

[Celebrate here]

If We're Just Tweaking...

We must be done!

[Celebrate here]

But then why is software still so hard to write?

Computing systems are doing so much:

Why can't we guarantee they work?

We believe that correctness is

the final frontier of programming.

How to Tackle Correctness?

- Mainstream languages:
  - Types (weak, but fully automatic).
  - Assertions (expressive run-time checks).
  - Testing (cannot show absence of bugs).
- Research languages and systems:
  - Model-checking (symbolic exhaustive testing).
  - Theorem proving (use computer proofs to show no bugs).
  - Advanced type systems (rich types express properties).

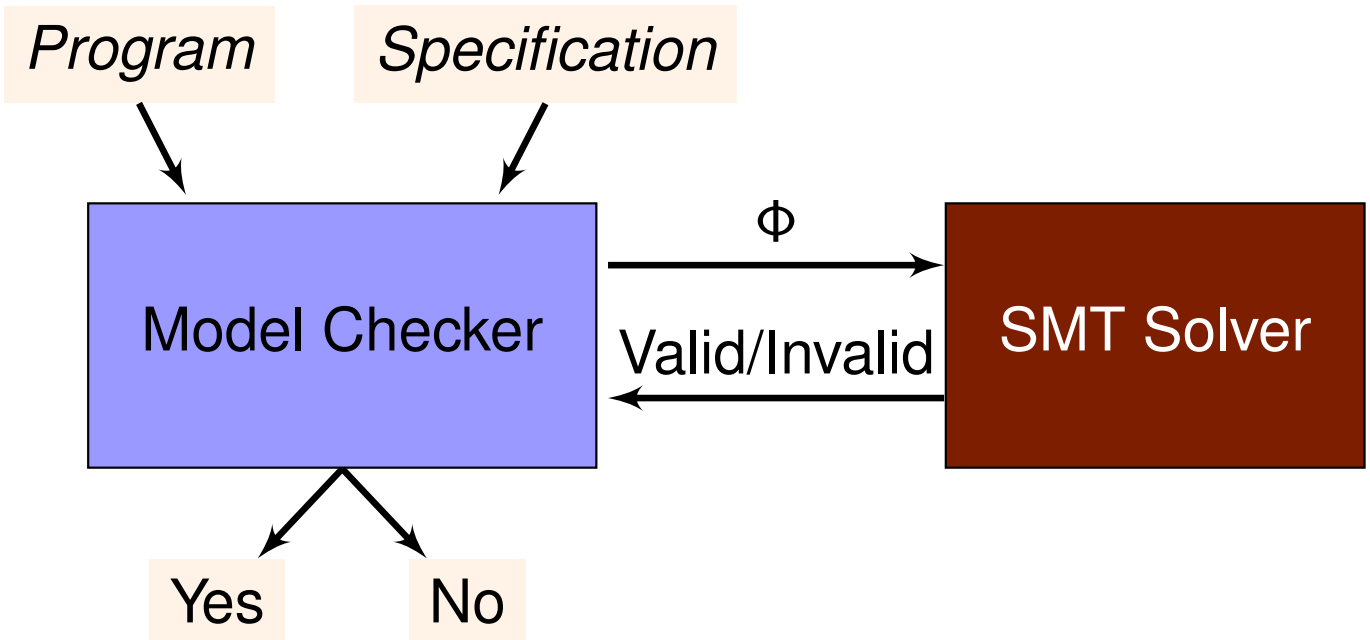
The Computational Logic Center at U. Iowa

- People:
  - Led by Prof. Stump, Prof. Tinelli.
  - Postdocs: Dr. Christoph Sticksel, Dr. Francois Bobot
  - Doctoral students: Mohammad Aziz, Frank (Peng) Fu, Tianyi Liang, Andy Reynolds, Harley Eades.
  - Master's students: Ruoyu Zhang.
  - Undergraduates: Angello Astorga.
  - Recent alumni: Dr. Duckki Oe (postdoc MIT), Dr. Garrin Kimmell (Kestrel Institute), Dr. Teme Kahsai (Skype).
- Collaborations:
  - CMU, NICTA (Australia), Minnesota, NYU, U. Penn., Portland State, JAIST (Japan)
  - Microsoft Research, Intel, Onera (France), Rockwell-Collins, Skype.
- Funding:
  - National Science Foundation, Air Force Office of Scientific Research, Intel, Rockwell-Collins.

Main focus: program verification via applied logic

Automatic Verification

- Goal: automatically prove properties about real code.
- Main techniques: model-checking and SMT.
- Important in industry, academia (Turing award 2007).



- KIND model-checker.
  - Prof. Tinelli, Dr. Jed Hagen (now at NASA), Dr. Ge (now at Two Sigma), Dr. Kahsai (now at Skype), Dr. Sticksel.
  - In use at Rockwell-Collins.

Verified-Programming Languages

- Existing languages use types to catch simple bugs.
  - No: `34 + "hi"`
  - No: `34("hi")`
- Fancier languages => fancier type systems.
  - JAVA, C#: generics.
  - HASKELL: polymorphism, type classes.
  - SCALA: mixins, implicits.
- Even more expressive: dependent types.
  - `list A ==> list A n`
  - `["a", "b", "c"] : list string 3`
  - `append : list A n -> list A m -> list A (n+m)`
  - Programs may contain proofs (e.g. that `n+m = m+n`).
- TRELLYS.
  - Prof. Stump, Harley Eades, Frank (Peng) Fu, Angello Astorga.
  - Building on previous work on GURU.

How You Can Learn More

- 22c:196:002, Lambda Calculus and Applications
  - Dr. Stump, MW, 2:00–3:15pm, English-Philosophy Building, Room 402.
  - Lambda calculus, including dependent type systems.
  - Foundation for functional programming, computer-checked proofs.
- 22c:188, Logic in Computer Science.
  - Dr. Sticksel, MWF, 11:30A–12:20pm, MacLean Hall, Room 105.
  - Propositional, predicate, temporal, and modal logics.
  - Knowledge representation and reasoning.
- Reading group on automated reasoning.
  - Fridays 2:00–3:30pm, starting August 31st.
  - Talk to us for more information.
- Talk to us or our students.

The Final Step

- Programming language evolution has plateaued.
- Building reliable software is a crucial challenge.
- Verification, verified-programming languages are the next step.
- The U. Iowa CLC is ready.
  - Logic.
  - Automatic Verification.
  - Verified-Programming Languages.
- We invite you to get involved!