Verification: What works and what does not?

Stephen A. Edwards
Columbia University
Verification at Columbia

- Luca Carloni et al. *model-checked some latency-insensitive hardware blocks*
  Do they faithfully implement synchronous semantics? Do they do so for all possible configurations?

- Steve Nowick et al. *develop aynchronous hardware components*
  Do they behave as advertised? Are they as efficient as they claim?

- My *SHIM language*
  I want static deadlock detection. I want to verify my implementation obeys the semantics.
Verification Successes and Failures

- Combinational equivalence checking
- Type checking in programming languages
- Model checking protocols
- SAT, BDDs
- Model checking real hardware designs
- Automatic software verification
- Theorem proving
How do we get to “cc -V 2”?

Aren’t we already there?

% gcc -Wall foo.c

% valgrind --tool=memcheck hello

% javac Hello.java

% ocamlc hello.ml

Programmers expect tools to behave like compilers

$O(n \log n)$ or die
The $24,000 question

What language constructs would make today’s and tomorrow’s verification algorithms practical?