Finding Bugs with Specification-Based Testing is Easy!

Janice Chan & David J. Pearce

```
1  import std::math
2  import contains from std::array
3
4  function max(int[] items) -> (int r)
5  // Cannot get max for empty array
6  requires |items| > 0
7  // Item returned must have been in items!
8  ensures contains(items,r,0,|items|):
9      //
10     int m = 0
11     //
12     for i in 1..|items|
13     where contains(items,m,0,i):
14         m = math::max(m,items[i])
15     //
16     return m
1
```

```
postcondition not satisfied
--> main::max([-3])
```

Compile ☐ Verification ☑ Check ☐ Counterexamples ☐ JavaScript **Examples:**

QuickCheck

> *"QuickCheck is a tool which aids the Haskell programmer in formulating and testing properties of programs. Properties are described as Haskell functions, and can be automatically tested on random input, but it is also possible to define custom test data generators"*
>
> *–Claessen & Hughes, ICFP'00*

- User-defined properties act as **test oracle**
- Demonstrated on **industrial-scale** projects
- Also available for Erlang, Java, C/C++ and more

Whiley

```
function max(int x, int y) -> (int z)
// result must be one of the arguments
ensures (x == z) || (y == z)
// result must be greater-or-equal than arguments
ensures (x <= z) && (y <= z):
...
```

- A language designed specifically to simplify **verifying software**
- Several trade offs e.g. **performance for verifiability**
  - *Unbounded Arithmetic, value semantics, etc*
- **Goal**: to statically verify functions meet their specifications

Evaluation (Benchmarks)

Name	Description	LOC
001_average	Average over integer array.	43
002_fib	Recursive Fibonacci generator.	17
003_gcd	Classical GCD algorithm.	40
004_matrix	Straightforward matrix multiplication.	139
006_queens	Classical N-Queens problem.	54
007_regex	Regular expression matching.	80
008_scc	Tarjan's algorithm for finding strongly connected components	201
009_lz77	LZ77 compression / decompression.	201
010_sort	Merge Sort	102
011_codejam	Solution for Google CodeJam problem.	118
012_cyclic	Cyclic buffer.	165
013_btree	Binary search tree with insertion / lookup.	172
014_lights	Traffic lights sequence generator.	64
015_cashtill	Simple change determination algorithm.	219
016_date	Gregorian dates.	88
017_math	Simple math algorithms.	193
018_heap	Binary heap data structure.	168
022_cars	Controlling cars on bridge problem.	54
	...	

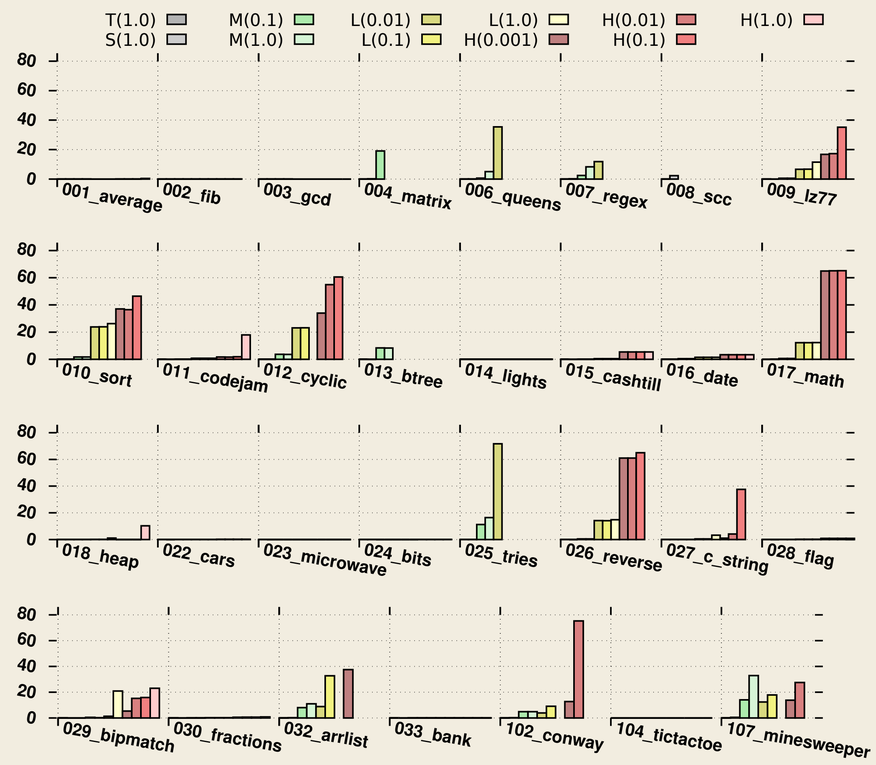
Evaluation (Sampling Rate)

- **(=1.0)** *every value taken*
- **(=0.1)** *every tenth value taken*
- **(=0.01)** *every hundreth value taken*

Evaluation (Scope)

	Ints	Arrays	Depth	Aliases	Rotation
tiny	⟨0...0⟩	⟨0...0⟩	⟨0...0⟩	⟨0...0⟩	⟨0...0⟩
small	⟨−1...1⟩	⟨0...1⟩	⟨0...1⟩	⟨0...1⟩	⟨0...1⟩
medium	⟨−2...2⟩	⟨0...2⟩	⟨0...2⟩	⟨0...2⟩	⟨0...2⟩
large	⟨−3...3⟩	⟨0...3⟩	⟨0...3⟩	⟨0...3⟩	⟨0...3⟩
huge	⟨−4...4⟩	⟨0...4⟩	⟨0...4⟩	⟨0...4⟩	⟨0...4⟩

Evaluation (Performance)



Evaluation (Precision)

http://whiley.org

@WhileyDave

http://github.com/Whiley