Programming research: a missed opportunity for secure and libre software?

Gabriel Scherer

Parsifal, INRIA Saclay
OCaml

July 3, 2019

Academic research

What is public research? How does it work?

- Pick a hard problem that we don't know how to solve.
- Come up with a (partial) solution.
- Evaluate it rigorously, compare with related work.
- Fully detailed presentation made **public**.

Software gets written. Some good. Mostly unmaintained prototypes.

Free Software / Academia collaborations

Many free software projects have hard problems to solve

- Compilers (Gcc, Clang)
- Systems research (Linux schedulers, etc.)
- Image processing (G'MIC, etc.)

What about **programming research**?

Free Software / Academia collaborations

Many free software projects have hard problems to solve

- Compilers (Gcc, Clang)
- Systems research (Linux schedulers, etc.)
- Image processing (G'MIC, etc.)

What about **programming research**?

Coccinelle for the Linux kernel.

Why3

Demo. (see demo code below)

```
let max_idx (a : array int) : int
  returns { best ->
    forall k. 0 <= k < length a -> a[best] >= a[k] }
=
  let ref best = 0 in
  for i = 1 to length a - 1 do
    invariant { 0 <= best < length a }
    invariant { forall k . 0 <= k < i -> a[best] >= a[k] }
    if a[i] > a[best] then
      best <- i;
  done;
  best
```

Program verification research (1/3): static analyzers

Static analysis tools: rule out entire classes of failures

Out-of-bound access, overflows, division-by-zero, use-after-free, etc.

Bug finding vs. spec writing.
(annotations are good for tools and humans alike!)

Success stories: Astrée (no runtime errors in the Airbus flight-control software), SLAM & Windows kernel, Facebook Infer.

Type systems: a special case, trying to remain simple to use.

Program verification research (2/3): Verified programming

Users write program and assertions / invariants,
tool translate them into goals for automated theorem provers.

Can prove more advanced properties.

Why3, Dafny, Spark/Ada, Frama-C.

Success stories: HTTPS stack in Everest, ProvenCore (Minix 3 variant).

Program verification research (3/3): Proof assistants

Users write a full mathematical proof, checked by the tool.
Can prove arbitrary results of mathematics or about programs.

Proof assistants: Coq, Agda, Isabelle...

Success story: SeL4, CompCert.

Adoption in Free Software:

Adoption in Free Software:disappointing!

FLOSS is lagging behind proprietary software on programming research adoption.

Faults on both sides: lack of time, unusable research prototypes, etc.

How to improve? (FLOSS contributors)

How to improve? (FLOSS contributors)

Easy: Safer language for new projects

How to improve? (FLOSS contributors)

Easy: Safer language for new projects

Medium: Seriously try to adopt static-analysis tools.
Provide feedback to researchers to scale their tools.

How to improve? (FLOSS contributors)

Easy: Safer language for new projects

Medium: Seriously try to adopt static-analysis tools.
Provide feedback to researchers to scale their tools.

Hard: stay informed about programming research.
Money: fund programmers to go to academic conferences.

How to improve? (FLOSS contributors)

Easy: Safer language for new projects

Medium: Seriously try to adopt static-analysis tools.
Provide feedback to researchers to scale their tools.

Hard: stay informed about programming research.
Money: fund programmers to go to academic conferences.

Money: fund some collaboration, or ask for joint financing.

Thanks

Questions?