CIFellows 2020-2021

Computing Innovation Fellows

Ben Greenman

BROWN UNIVERSITY

Type systems, logics, and modeling languages are powerful tools for writing formal specifications. Specifications can help programmers validate their designs and even synthesize components — provided that the specifications are correct.

But specifications may not model the intended behavior.

PROBLEM: Specifications can be wrong.

To address this general problem, we need to study faulty specifications and develop actionable plans.

Q1: In what ways? Q2: What can we do about it?

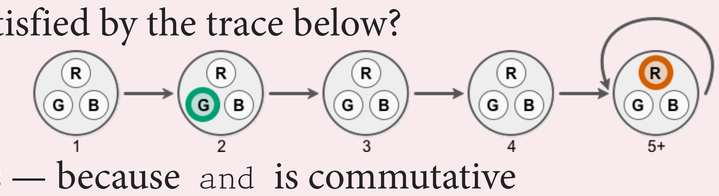
I have been studying these questions in several contexts using a variety of research methods.

LTL Formulas

RQ. Do users understand linear temporal logic?

Surveys, Talk-alouds, Crowdsourcing

Q. Is the formula `eventually Red and eventually Green` satisfied by the trace below?



A. Yes — because `and` is commutative

Test instruments, Code book of errors

Alloy Models

RQ. What misconceptions do programmers face?

File snapshots, Grounded theory

Language levels: functional, relational, temporal

Algebraic Properties

RQ. Can learners write correct instances?

Wheat and Chaff specifications

Incorrect Example:
transitivity vs. reachability

```
A = a + b + c
r = a->a + b->b + c->c
  + a->b + b->c + c->a
```

Data-driven chaffs

Type Interfaces

RQ. How well do languages help debug wrong types?

Mutation analysis, Experiment design
"The Rational Programmer"

TRP method, Evidence for design choices

Thanks to the CIFellows program, I have gained experience with a broad set of research methods that I can use throughout my career.

CRA Computing Research Association

CCC Computing Community Consortium Catalyst

NSF