Project Summary

Overview

Property-based testing (PBT) is an advanced software engineering methodology where users write executable formal specifications of system components and an automated harness checks these specifications against many automatically generated inputs. The bug-finding power of PBT stems from its ability to express rich specifications that exercise a wide range of system behaviors—both expected and unexpected—with minimal user guidance. From its roots in the Haskell QuickCheck library, PBT has become the testing method of choice across much of the functional programming community, and it is making inroads into industrial practice at companies such as Amazon, Volvo, Stripe, Galois, and IOG. The goal of this proposal is to accelerate this transition by addressing key research challenges in PBT.

What are these challenges? In an ongoing need-finding study of PBT at Jane Street, a Wall Street trading firm which uses its software technology as a competitive advantage, we find developers enthusiastic about its *usefulness* but frustrated with its *usability*. Indeed, this study has identified usability as an issue in both major aspects of PBT methods—*specification* of properties and *generation* of random inputs. Moreover, it reveals other challenges around another aspect of PBT that has so far received less research attention: how programmers *interact* with their programming environment during the testing process.

To address these challenges and establish PBT as a mainstream testing method, insights from two distinct research areas must be brought to bear. On one hand, PBT itself is grounded in domain-specific languages and formal methods—topics traditionally associated with Programming Languages and included in the SHF program under CISE-CCF. Usability, on the other hand, is the domain of Human-Computer Interaction; its center of mass in CISE is IIS's HCC program.

Keywords: Programming languages; human-computer interaction; property-based testing; usability.

Intellectual Merit

The project will advance knowledge along four interconnected axes. First, it will establish a firm ***foundation*** for HCI-informed research on PBT, supplementing past and ongoing user studies with broader surveys of PBT across the software industry, real-time observations of developers interacting with PBT, and a novel cognitive theory of PBT. Second, it will explore a new abstraction for random input ***generation***, "reflective generators," that enables a range of use cases—generating inputs satisfying validity conditions, reducing test-cases for easier debugging, mutating inputs to explore program behavior, and tuning distributions based on examples or coverage. We will also explore automatic generator construction and develop benchmarks for PBT tools. Third, it will offer developers more usable ***specification*** tools, including a language for temporal properties over program traces, mixed-initiative interactions for defining properties, automatic model-based testing of modular abstractions, and techniques for explaining properties. And fourth, it will develop new tools for effective ***interaction*** between developers and their tests, including tools for visualizing and manipulating generated data distributions, simplifying test-cases, and saving failing test-cases as understandable unit tests, culminating in a VSCode extension for PBT.

Broader Impacts

A final thread of activity, coordinated with the rest and integral to the project's aims, will be to drive the ***diffusion*** of PBT tools and methodologies from academia into industry through targeted engineering effort and a broad range of educational activities. We will support and strengthen existing open-source PBT tools in popular programming languages and enrich them with well engineered research products from the four themes described above; and we will develop materials for teaching industry programmers how to identify high-leverage PBT opportunities and curricula for teaching mature and powerful PBT practices to undergraduate and masters students.

Work on the project will include and elevate undergraduate researchers, including some from diverse backgrounds that will be identified and supported through cooperation with a recently funded NSF-REU effort at Penn. These undergraduates will work closely with the PIs, Ph.D. students, and staff engineer.

The ultimate goal, through novel research, education, publications, and open-source tools, is to advance the state of the art in property-based testing and put it on every software developer's testing toolbelt. Better testing, in turn, will lead to software systems that are more robust and less expensive.