Personal Statement

In the decade since I first started programming, I've managed to learn a lot about myself and what motivates me to do the work that I do. In particular, I have come to understand how important improving the lives of others is for me. Because of this, **I've chosen to orient my life in computing around impacting others in my research and beyond.**

As an aspiring young researcher, I was drawn immediately to the field of programming languages. When I learned to program, I found myself most interested in languages like Haskell and Rust which offer the programmer a variety of safety guarantees. As I programmed more, the extent to which these guarantees were necessary became apparent as the complexity of software continued to grow. While many languages today offer some sort of guarantee, they are often too weak to deal with modern software engineering requirements or they simply fail to take advantage of domain-specific knowledge. These are the problems that interest me because by improving our programming languages, I can impact the discipline of software engineering as a whole, and can help make programs more correct and secure. However, these concerns of safety and security are rarely present in early computing curricula, and this understanding is something I firmly believe should be taught to students.

To that end, in high school, I organized the first ever high-school-run national computer science competition, HSCTF, with a small team of friends. The focus of the competition was on exposing students to the computing skills necessary to consider the security and correctness of programs — even in the face of buffer overflows or cross-site scripting. We also hoped to address other potential deficiencies and included a variety of problems, covering topics such as graphs and dynamic programming. Knowing these sorts of competitions were typically accessible only to the most advanced students, we made a concerted effort to reach out to high schools with students that we believed would otherwise not receive this degree of exposure — especially local schools known for their diversity. Our efforts were greatly rewarded, and our inaugural competition had over 2,700 participants. Many of these participants were inspired by our success and went on to organize their own competitions.

In fact, HSCTF has inspired hundreds of such competitions run by high schoolers around the world. Today, you can compete in a high-school-level CTF nearly every weekend. After starting University, an HSCTF participant reached out to me for help running a new competition called sCTF which gave me the opportunity to continue this effort. With sCTF, we set continued exposure as our goal, and thus sought to run four competitions per year. Like HSCTF, these competitions were a resounding success — each reaching over 3,000 participants. After the first sCTF, I was invited to speak at a local high school in New Jersey about my experience running these competitions, working in and studying computer science, and the available career options in computing fields. This experience really made me feel that the work I had done on these competitions was genuinely helping to make computer science more accessible to those traditionally kept from it. It also gave me the opportunity to talk to some of the students who were participating in these competitions directly.

While there, I spoke to two classes of students in AP Computer Science and I was amazed at the diversity I saw — there were students of all races and genders, united by the fun and excitement of a competition that I helped create. Speaking to them helped me realize that I could more clearly incorporate my own interests into the next sCTF, and so I made a category of problems focused on some of the foundational topics from programming languages research, including parsing, AST evaluation, SMT solving, and program verification.

Relevant Background

Intellectual Merit. As a high school student, I was fortunate enough to have the opportunity to take a number of computer science courses that covered the whole of the introductory curriculum at the University of Massachusetts, Amherst (UMass). As a result, I was able to do well in classes while still investing a great deal of energy into other intellectual pursuits like working on my open source software, and learning about programming languages. By the time Fall Break rolled around, I had begun self-studying Types and Programming Languages[1] and reaching out to faculty who worked on programming languages and systems. It took me until the Spring to get a response, but when I did, I quickly got to work with Prof. Arjun Guha, an assistant professor in Computer Science at UMass.

Immediately, I was thrown into the realm of the unfamiliar — even in spite of my extant efforts to study the fundamentals of the field. My work began with a static analysis for a declarative system configuration language called Puppet. I first approached the problem trying to leverage predicate transformers, but quickly learned that they were not an appropriate solution. I then built the analysis using Z3, a high performance SMT solver, instead. I continued working on this project over the summer as an REU by extending the static analysis tool with a program synthesis component. Largely unfamiliar with the state of the art, I studied a number of papers on synthesis by sketching[2]. We then developed a small sketching language with a prototype of the tool built around it, and when we learned this approach would not scale, we adapted our technique to incorporate a better search procedure.

My experience studying and working on program synthesis interested me in its integration within languages to support new abstractions or to provide new methods of utilizing and interacting with software and data. At the end of the summer, **I attended the International Conference on Functional Programming on a Travel Scholarship from the Programming Languages Mentoring Workshop.** While there, Prof. Rastislav Bodik of the University of Washington gave a keynote talk about program synthesis where he spoke in depth about its future and potential applications. Given my newfound excitement about synthesis, this talk was incredibly invigorating and further reinforced the appeal of synthesis in my mind. In the end, the synthesis tool we developed was hard to formalize, and we set it aside. This experience nevertheless fomented my passion for programming languages, exposed me to a wealth of new knowledge, and solidified my interest in a research career.

We continued working on the project into the Fall, and in the Spring, **our paper, *Rehearsal: A Static Verification Tool for Puppet*, was accepted to the ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI).** It covered the whole static analysis featuring a ready-to-use artifact and proofs of soundness and completeness. At the conference, **we were awarded the Distinguished Artifact Award for the ease of use and extensibility of our tool.** Afterward, I returned to UMass for another REU where I once again set out to bring program synthesis to Puppet — this time with a different angle, inspired by a paper I saw presented at PLDI. By the end of the summer, I had a prototype tool that could synthesize updates to Puppet programs based on a subset of shell commands. I'm now continuing this work during the semester with the hopes of publishing an additional conference paper.

[1] Types and Programming Languages is a popular textbook by UPenn professor Benjamin Pierce.

[2] Sketching is a program synthesis technique popularized by MIT professor, Armando Solar-Lezama.

Broader Impacts. Over the years, I've volunteered as a mentor for a number of educational programs for youth in computer science. In high school, I organized a series of workshops on Lego robotics for fourth and fifth graders in conjunction with the public library and my school's robotics team. Twice a year, we offered a ten week program with classes once a week teaching programming and robotics. Our classes were well-balanced in gender and the students loved the opportunity to be creative — especially during one of the classes where they built sumo-robots and tried to come up with the best designs for pushing one another out of the ring. Similarly, in university, I volunteered for a workshop for Eureka!, an educational program focused on exposing young women over the course of high school to a variety of topics in Science, Technology, Engineering and Math. Over the summer, we ran a week-long daily workshop to teach Scratch to girls in their first year of the program. Like the Lego robotics class, the focus was on exploration, and as a mentor, I worked one-on-one with the participants. Volunteering in these programs was incredibly rewarding for me, and helped motivate my interest in academia by helping to ignite a passion for teaching.

I am incredibly fortunate to have had the opportunity to see teaching both in an outreach capacity and a traditional classroom setting. The competitions I ran helped me understand how curriculum development can positively affect students en masse by providing a ready example of how what is assigned can affect what students learn. Meanwhile, the workshops I've volunteered for have enabled me to work one-on-one with students and understand intimately the sort of impact I can have on a student's success when teaching. I've also been working as a teaching assistant for Programming Methodologies, one of the core classes in my school's computer science curriculum, and this experience has also furthered my love of teaching. It has given me the opportunity to run both discussion sections and office hours where I've been able to work with students in a more traditional classroom setting and one-on-one respectively. I realized that guiding a student towards the solution to a problem that is difficult for them gives me the same rush as that of solving a problem that is difficult for me, and this feeling keeps teaching fun and exciting. These experiences together have ultimately helped me understand myself and the sort of work that motivates me.

Future Goals

I have grown considerably as a result of my experiences in research and outreach, and I believe that pursuing an academic career will best enable me to continue these experiences into the future. In particular, **I would like to be a tenure-track professor at a public university because I deeply value the goal of providing an education to people of all backgrounds.** The allure of such a position comes from the intersection of all my experiences in both research and outreach. My research has presented me with the opportunity to challenge myself intellectually, and advance the state of the art while making software engineering more accessible to people of all skill levels. My outreach has presented me with the opportunity to share my passion for computer science with others and to make the field as a whole more accessible to people of all backgrounds. My teaching has presented me with the opportunity to watch and help others grow and to prepare my students for a world increasingly dependent on computing. Of all the things I've been fortunate enough to do in my life, I cannot imagine anything more fulfilling than having the freedom to pursue these passions together, and I believe graduate school and the NSF Graduate Research Fellowship would thoroughly enable me to do so.