STATEMENT OF PURPOSE

Chenhao Gao **Zhejiang University**

Motivation

My research interests include type systems, functional programming, and programming language design and implementation. My long-term goal is to develop expressive and intuitive programming languages with strong guarantees, making programming engaging and accessible to everyone.

My entry point into this field was a third-year course, *Introduction to Theoretical Computer Science*, where I discovered that, in addition to the exhaustive specifications for programming languages, it is possible to specify a language both formally and concisely. This sparked my interest in exploring advancements beyond the simple example languages taught in class. With no advanced courses available, I pursued independent study in programming. As I delved deeper, I became increasingly fascinated by the sophisticated theoretical foundations and the harmonious integration of elegance and engineering in type systems and functional programming languages. Since then, I have actively sought research opportunities in this field.

My path to research was not straightforward during the first two years of my undergraduate studies. I lacked clarity about my research interests. In China, there is a strong trend for students to join labs early and pursue popular academic topics, and I was no exception. However, as the initial novelty wore off, I realized my true interest lay elsewhere. Rather than merely following prevalent trends or working on topics predefined by others, I wanted to develop my own research perspective. Motivated by this clarity, I began exploring fields that inspired me, venturing beyond my established foundations to embrace new challenges. Fortunately, programming languages turned out to be the perfect fit for me.

Academic Background

The first course I took related to programming languages, after *Introduction to Theoretical Computer Science*, was *CS 242: Programming Languages* from Stanford. It provided a solid foundation in lambda calculus and type systems. Some of the assignments I completed are available here. Among the many thought-provoking assignments, my favorite was implementing an interpreter for a typed lambda calculus language, where I found it particularly rewarding to bring theoretical constructs to life in a functioning system. Overall, this course equipped me with the necessary background knowledge and bridged the gap between theory and practical application.

Building on this foundation, I further deepened my understanding through the Compiler Principles course in 2024. I implemented a compiler in OCaml independently, achieving a comprehensive understanding of both compilers and functional programming. Through dedicated efforts, I earned a perfect score on all the labs of the compiler project. The expressive type system and pattern-matching capabilities of OCaml made the development process intuitive and error-resistant compared to imperative languages like C++, where debugging crashes often consumed significant time. This experience strengthened my appreciation for functional programming and robust type systems, while teaching me the methodology for designing and implementing compilers.

Research Experience

Despite the limited time available to apply for further research opportunities and the underexplored nature of programming languages in China, I proactively contacted professors to seek potential research projects. My initial opportunity was offered by Associate Professor Peisen Yao at Zhejiang University,

where we worked on accelerating the parallel solving of the Satisfiability Modulo Theories (SMT) solver Z3, with a particular focus on solving string constraints. We aimed to incorporate a machine learning model to augment the solver's decision-making capability, but faced three primary challenges:

1. **Understanding Z3**: Z3 is a large open-source C++ project with over 450,000 lines of code, sparse documentation, and intricate internal representations. To navigate these challenges, I synthesized all available information from manuals, papers, and documentation to understand key functions and adapt them to meet our requirements.

2. **Integrating Machine Learning**: Incorporating a machine learning mechanism into Z3 was challenging because most models and frameworks are implemented in Python, not C++. To address this, I trained and evaluated the model in Python and wrote scripts to enable inter-process communication between Z3 threads and the Python interpreter.

3. **Characterizing Solver States**: Representing the state of the solver for input to the model required identifying both static features and features derived from look-ahead and look-back heuristics. Since most relevant statistics were not immediately accessible in Z3, I developed custom code to gather and output the necessary data as input for our model.

The integration of machine learning into SMT solving marks a revolutionary shift from traditional handcrafted heuristics to efficient, data-driven models. Our approach improved solving speed by 1.39× and 1.64× for different backend configurations. Parallelism in string SMT solving is crucial for reasoning about string manipulation programs, which underpin inter-program communication, security policy enforcement, and various security-dependent fields. I independently implemented and evaluated the entire project, designing processes to modify, test, and integrate improvements into the SMT solver. Our findings are detailed in a paper under review at ISSTA, where I am the first author.

Following this, I joined an internship program supervised by Associate Professor Magnus Madsen at Aarhus University, where I am actively contributing to the development of the Flix Programming Language. It is immensely rewarding to contribute to a real-world functional programming language. My current responsibilities include integrating functionalities such as completions and code actions into the Language Server Protocol (LSP) to enhance the user experience, forming the foundation for an in-person internship phase in December. This experience has significantly improved my ability to write well-documented and readable code, a critical skill for effective collaboration on complex projects. It has also required me to adopt a user-centric perspective to design LSP features that streamline and accelerate everyday programming tasks. Promoting features from prototype to maturity has been a deeply satisfying process.

Career Goal

Building on my prior research and development experiences, I am eager to advance to the next level in my journey as a researcher in programming languages. My ultimate goal is to develop intuitive tools and methodologies that make programming engaging and accessible to everyone.