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 research aspiration is to develop expressive and intuitive programming languages with strong guarantees, making programming engaging and accessible for 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, an alternative way exists 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 Languages. As I delved deeper, I became increasingly fascinated by the sophisticated theoretical foundation and the harmonious integration of elegance and engineering in type systems and functional programming languages. Since then, I have been pursuing research opportunities in this field.

My path to research was not straightforward in the first two years of my undergraduate study. I lacked clarity about my research interests. However, there is a strong trend for students to join the labs early and pursue popular academic topics in China, and I was no exception. Yet, as the initial novelty wore off, I realized my true interest lay elsewhere. Rather than merely following the prevalent trends or working on topics predefined by others, I desired 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 related to Programming Languages after *Introduction to Theoretical Computer Science* was *CS 242: Programming Languages* from Stanford, which provides a solid foundation in lambda calculus and type systems. Some of the assignments I completed are available here. Among all the thought-provoking assignments, my favorite was the interpreter for a typed lambda calculus language, where I found it rewarding to make theoretical constructs come alive in a functioning system. Overall, this course equipped me with the necessary background knowledge and bridged the gap between theoretical knowledge and practical application.

Building on the foundation laid in the Programming Languages course, I further deepened my understanding through the Compiler Principles course in 2024, where I implemented a compiler in OCaml independently to attain a comprehensive understanding of both compiler and functional programming. Through my dedicated efforts, I achieved a 100 score on all labs of the compiler project. The expressive type system and pattern-matching ability offered by OCaml made the developing process intuitive and error-resistant, compared to imperative languages like C++, where I just kept finding out why my program crashed. This experience deepened my appreciation for functional programming and its robust type systems and taught me the methodology to design and implement a compiler.

Research Experience

Despite the limited time to apply for further research opportunities and the under-explored nature of Programming Languages in China, I proactively contacted professors to find any potential research opportunities. My initial research opportunity was offered by Associate Professor Peisen Yao at Zhejiang

University, where we worked on accelerating the parallel solving for a Satisfiability Modulo Theories (SMT) solver Z3, with particular emphasis on solving string constraints. We tried to incorporate a machine learning model to augment the solver's decision-making capability, but there were three primary challenges to achieve this:

1. Understanding Z3, a large open-source C++ project with over 450,000 lines of code, required navigating its intricate internal representation, sparse documentation, and uncommitted functions. To resolve this challenge, I combined all available information sources from manuals, papers, and documentation to understand the meaning of relative functions and adapt them to meet my requirements.

2. Incorporating a C++ machine learning mechanism into Z3 is challenging, as most machine learning models and frameworks are implemented in Python, not C++. I managed to train and evaluate the model in Python and write scripts to achieve inter-process communication between every Z3 thread and the Python interpreter.

3. Characterizing the state of a solver, which will later become the input of the model, is a non-trivial endeavor. Our method is to identify a set of static features, as well as features that involve look-ahead and look-back heuristics, to represent the state of the solver. Since most of the statistics are not immediately available on Z3, I developed code to gather and output them as the input for our model.

The introduction of machine learning to SMT solving is a revolutionary shift from traditional hand-crafted heuristics to efficient and powerful data-driven models. Our method enhanced solving speed by 1.39× and 1.64× for different backend configurations. Parallelism in string SMT solving is crucial for reasoning string manipulation programs, which are fundamental for inter-program communication, enforcing security policies, and numerous other security-dependent fields. I independently implemented and evaluated the whole project, designing processes to modify, test, and integrate improvements to the SMT solver. Our findings were detailed in a paper under review at ISSTA, where I am the first author.

Following this, I am participating in an internship program supervised by Associate Professor Magnus Madsen from Aarhus University. I am actively developing the Flix Programming Language! It's fascinating to contribute to a real-world functional programming language. I am responsible for integrating functionalities such as providing completions and code actions into the Language Server Protocol (LSP) to enhance users' coding experience, an initial phase for the in-person internship in December. This experience has significantly enhanced my ability to write documented and readable code, which is essential for collaborating effectively on complex projects. This project requires me to adopt a user-centric perspective to design satisfying LSP features that streamline and accelerate users' everyday programming experience. It's a great pleasure to promote any handy feature from prototype to maturity.

Career Goal

Building upon my prior endeavors, I am eager to progress to the next level to establish myself as a researcher in Programming Languages, developing intuitive tools and methodologies that make programming engaging and accessible for all. This is why I am applying to graduate programs.