Michael Fortunato's Statement of Purpose

Since 2018, my goal has been to pursue graduate studies in Computer Science, with a focus on either complexity theory or systems programming problems. After three years of experience as a systems developer in C and assembly for the database company InterSystems, I have honed my interests and believe I am well suited to succeed in Stanford's masters program studying system's programming. Stanford's lively state of systems development, particularly the research of Dr. Christos Kozyrakis and Dr. David Mazières, including their paper Syrup[1], which gives application programmers more control over the scheduler, and RAIL[2], which is a collection of techniques to reduce read tail latency on flash in the presence of writes, exemplify how researchers at Stanford recognize that the next generation of system's development will consist of giving application programmers more control over resources and will consist of developing increasingly sophisticated memory management techniques.

Very broadly, my academic interests are in exploring systems programming problems through a theoretical lens so that they can be better understood, and I am particularly interested in doing this for memory management and compiler optimization passes. For instance, I would like to contribute to recent improvements in memory management scheme aiding the translation lookaside buffer, such as the Mosaic project, which is able to deal with increasing RAM size by employing virtually large pages, all while keeping physical pages small. I know from my job developing a database engine the importance of large pages for faster computing in certain applications (such as databases). I am interested in expanding upon this research, with an eye towards seeing how these techniques do in NUMA environments. Another area I am deeply interested in is formalizing the cost models of compiler optimization passes. Historically, compiler optimization passes have been based on heuristics. By mathematically expressing the cost and benefit of these optimization passes, we will have more confidence in these passes, and may be able to automate their construction. The Tensor Algebra Compiler (TACO), is a project that seeks to formalize cost models of sparse tensor multiplication. The library writers, by better formalizing sparse tensor multiplication in a computer setting, created a compiler that generated efficient multiplication code, whereas in the past the code was written manually and highly specialized to the value of the tensor. These are the sorts of initiatives in which I want to do research and contribute, where systems problems are formalized and solved through theory.

---

[1] Kostis Kaffes, Jack Tigar Humphries, David Mazières, Christos Kozyrakis
Proceedings of the ACM SIGOPS 28th Symposium on Operating Systems Principles (SOSP)
October 2021

[2] Heiner Litz, Javier Gonzalez, Ana Klimovic, Christos Kozyrakis
ACM Transactions on Storage

Since completing my undergraduate degree, my three-year career as a developer at InterSystems has uniquely prepared me to do theoretical research in the area of systems programming. As a new graduate, I envisioned and created the HLv2 Data Profiling Tool (DPT) for InterSystems, a HL7v2 message analysis application designed to quickly process hundreds of thousands of HL7v2 messages at once. The DPT has a multi-process analysis algorithm for performance, has a highly optimized database schema for analytic queries, supports a plugin validation design, and has a modular architecture for good separation of concerns, along with all the other components one would expect from an enterprise grade application, like an installer, and a good CI/CD pipeline. After an internal trial with our customers, the application proved successful, and soon a formal full-time team (Time To Value) was assembled for adding more features to it, of which I became the technical lead.

The DPT's modular architecture, which emphasized composition over inheritance and aggressively enforced the use of interfaces, became crucial once we received feedback from our customers. Large Health Information Exchange customers valued the DPT because its ingestion algorithm was able to thoroughly analyze hundreds of thousands of messages quickly (under 15 minutes, a first in the industry), and the DPT's resultant model was designed for very fast (sub 200 ms) arbitrary queries, such as "how many times did my PID-5 field occur with invalid names (as I have defined invalid) amongst this hospital". They quickly saw the DPT not as an on-line service that updated a live model, but rather a data modeling tool that was snapshot based, and we were able to turn the Data Profiling Tool into a snapshot based modeling application over the period of 6 weeks during the summer of 2022, all without blocking other developers on our team from developing new DPT features–a testament to its modular design.

The Data Profiling Tool was finally released to customers this Spring 2023, and was voted by our customer's as "Most Innovative and Likely To Use" in June at our company's annual conference, Global Summit.

After the Data Profiling Tool was released, I was invited to join the company's Kernel group as its youngest developer. InterSystems is a database company used by major organizations such as Epic and the Veterans administration. The Kernel team, which is responsible for developing the database kernel and ObjectScript language, is a highly research oriented systems programming group, working exclusively in C and assembly, and the majority of its 12 members are career system programming veterans, with experience at DEC and Intel and/or advanced degrees. In order to join the team, I self-studied the code base on weekends, and one weekend implemented the `for x in list` language construct, as seen in Python, in ObjectScript. To join this group, explicit

approval is required by our CEO and founder, Terry Ragon. I have already learnt much from these members, and they have encouraged me to pursue an advanced degree. I am grateful for their mentorship.

On this team, my responsibilities are to research new ways to improve the ObjectScript runtime performance. Given the team's research oriented nature, we share papers with each other on the latest in database management and compiler theory and are expected to read them and provide thoughts, and even prototypes. Specifically, we are working on performing optimization passes for ObjectScript, like for-loop fusing/unrolling, emitting SIMD instructions, and doing monomorphization in place of polymorphism. On the database end we are exploring the use of Log Structured Merge Trees as a complement to our B+ trees in order to support higher write throughput.

In conclusion, my targeted interests in Systems Programming, combined with my early career industry experience in low-level programming and theoretical inclinations, make me both a well-suited and a unique candidate for this master's program. I am eager to further hone my skills and knowledge in preparation for doctoral studies. Along the way, I plan to contribute to the next generation of open systems software, and am a particular fan of the Theseus OS project. Thank you for taking the time to read my application.