

Shrikanth Sudhersan

✉ ssudher@ncsu.edu  linkedin.com/in/ssudher  github.com/ssudher

📍 2800-23 Brigadoon Drive, Raleigh, NC - 27606 📞 919-345-5968

EDUCATION

North Carolina State University, Raleigh, NC, USA

August 2019-May 2021

Masters in Computer Science, GPA : 3.93/4

Coursework: Operating system, Parallel systems, Game-Engine foundations, Statistics, Algorithms, Internet Protocols

Amrita Vishwa Vidyapeetham, India

August 2012-May 2016

Bachelor of Technology in Electronics and Communication Engineering, Distinction

WORK EXPERIENCE

Synopsys, North Carolina, USA

May 2020-August 2020

Software Intern

- Worked on enhancing the existing product line by integrating a C++ based proprietary distributed programming library.
- Designed programs to stress-test the library in a distributed environment to identify potential failure scenarios.
- Built custom programs to emulate the type of jobs in the product to gauge performance and resilience of the library.
- Redesigned the code to improve the cache reuse and hit-ratio which boosted the performance by 20%.
- Collected data and prepared documentation to identify best practices for fault-tolerant usage of the distributed library.

Honeywell, Bangalore, India

January 2016-April 2019

Senior Software Engineer

- As a part of the research and development team, conceptualized a unified framework using Python and C# to generalize and automate development of aerospace software. This reduced wastage induced through human errors by 40%.
- Programmed a method to parse and understand the source code(C/C++) to build a custom code comparison module.
- Developed a chat-assistant that understands user intents and gives useful insights into specific industry jargon. Leveraged Python's (NLTK) Natural Language Tool-Kit, C# and SQL database to devise the framework.
- Programmed a Python application to automate the development of test-cases for several categories of requirements that eliminated manual effort and human prone mistakes. It saved 30% time in completing the program.
- Recognized as the 'Outstanding Achiever' of the year 2017.

TECHNICAL SKILLS

Language/Database: C, C++, Python, OpenMP, OpenMPI, OpenAcc, CUDA C, SQL, MongoDB

Tools/OS: JIRA, GitHub, GitLab, Wireshark, Linux, MacOS, Windows, Hadoop, Spark, MapReduce, GDB

PROJECTS

Game Engine  | C++


August 2020-Current

- Designed and implemented a game engine from scratch. Used Inheritance, object-oriented design concepts, multi-threaded game loop architecture and SFML library to render objects onto the screen.
- Incorporated multi-player support using ZeroMQ with custom serialization & de-serialization methods from scratch.

XINU Operating System  | C

August 2019-December 2019

- Implemented memory virtualization through demand paging using swapping algorithms: Second Chance and Aging.
- Designed a robust scheduler(exponential-distribution, linux-like) and lock(reader/writer) with priority-inheritance.

Point-to-Multipoint File Transfer  | Python, Socket programming

August 2019-December 2019

- Using the underlying principles of UDP as the transport layer protocol implemented a multi-threaded program that uses ARQ methods to achieve reliable file transfer from one sender and multiple receivers.

Bit-Torrent  | Python, Socket programming

August 2019-December 2019

- Programmed a peer to peer file sharing model among distributed peers and compared it with centralized server model.

Cache Coherency Simulator  | C++

August 2019-December 2019

- Implemented bus based cache-coherency protocol MSI,MESI,Dragon for a shared L1 cache in a multicore processor.
- Simulated and analyzed the protocols for their performance by varying cache size, block size and associativity.

Simplified Custom Message Passing Directives  | C, OpenMPI, pthreads

January 2020-May 2020

- Implemented the basic MPI directives such as Sendrecv, Barrier, Init, Gather and Finalize using producer-consumer pthread functions and socket programming in C.

Accelerating Graph algorithm  | C, OpenMP

August 2019-December 2019

- Designed a memory-efficient adjacency lists for building Compressed Sparse Row (CSR) using Radix sort