Zichao Zhang

Phone: (929) 398-1879
E-mail: zbz5068@psu.edu
Website: http://zichaozhang.com

RESEARCH INTERESTS

Programing languages and software security

EDUCATION

August 2015-Present **The Pennsylvania State University** University Park, PA

B.S. in Computer Engineering with University Honors, December 2018
GPA: 3.98/4.0

- Relevant Coursework: Programing Languages Concepts, Computer and Network Security, Data Structures and Algorithms, Operating Systems, Computer Architecture, Microprocessors and Embedded Systems, Computer Vision

RESEARCH EXPERIENCE

Laboratory of Prof. Danfeng Zhang University Park, PA

August 2017 – Present Research Assistant - Penn State Department of Computer Science and Engineering

Bayesian Reasoning for Automatic Security Mediation Placement

- Collected benchmarks
- Wrote programs to automatically evaluate the framework

Exploring Fast Algorithms for Dyck-CFL-Reachability

- Designed path finding algorithm for Context Free Language reachability
- Achieved order of magnitude speedup compared to the state-of-the-art

TEACHING ASSISTANT EXPERIENCE

Fall 2018 CMPEN472: Microprocessors and Embedded Systems The Pennsylvania State University

- Held 3 hours weekly office hour
- Graded assembly code

PROGRAMING SKILLS

Have recent experience with OCaml and Scheme; have some experience with C/C++, Java, Python, Perl, MATLAB, Verilog and Assembly

AWARDS AND HONORS

Spring 2018 REU Scholarship
Spring 2017 Evan Pugh Scholar Award
Spring 2016 President's Freshman Award
2016 – 2018 Penn State Schreyer Honors College

PROJECTS

Golang Channels Implementation in C

- Implemented a synchronized channels API in C
- Supported blocking/non-blocking channel send and receive

User-space Synchronization and Thread Library

- Used x86 assembly and C to implement several synchronization primitives such as mutexes, condition variables, semaphores and readers/writers locks
- Built a user-space thread library based on the kernel threads

Malloc Implementation

- Implemented a dynamic storage allocator
- Used single linked list, double linked list and binary search tree to store different sized free blocks to achieve speed and efficiency
- Achieved 100% on both space utilization and throughput