Gao Zhiyuan

+86 13262224615
alapha23@gmail.com

Education

- **National Cheng Kung University** — Taiwan
  B.S., Engineering Science — *Sep. 2015 - May. 2017*
- **Seoul National University** — Korea
  Undergraduate, Computer Science and Engineering — *Sep. 2017 - ongoing*

Work Experience

- **Software Platform Lab** — Seoul National University
  Intern — *Mar. 2019 - ongoing*
    - Part of a group working on distributed dataflow system – Nemo, an Apache Incubator project

- **SUSE** — Beijing, China
  Intern — *Oct. 2018 - Jan. 2019*
    - Part of the QA team working on openSUSE and SUSE Linux Enterprise
    - Add QA test cases with Perl Programming on openQA, with test cases version control and bug follow-ups

Publications

- **Serverless Computing: Pitfalls and Solutions**
  Korean Computer Congress 2019 — *First Author*
- **Lambda Executor: extend Apache Nemo with serverless functions**
  Korean Software Congress 2019 — *First Author*
- **A third-author paper rejected by EuroSys 2020**
    - A hyrbid system for dealing with bursty input rates in stream processing applications, which dynamically creates VMs and serverless containers.
    - Actively participated in paper writing, research experiments and evaluation, algorithm design regarding VM scaling, and engineering the dataflow framework.

Projects

- **Incubator-nemo Lambda Executor** — Apache Software Foundation
  Google Summer of Code — *May. 2019 - Sep. 2019*
    - Migrated executor for batch applications to AWS Lambda functions.
    - Enabled executor on lambda functions' lifecycle management. Project only tested on single-stage stateless applications.

- **SnuPl/1 compiler** — Sep. 2017 - Jan. 2018
    - Programmed and debugged full frontend of the compiler.
    - Implemented compiler code generator for x86 assembly.

- **Weighted Round Robin Scheduler** — Mar. 2018 - Jul. 2018
    - Worked on the WRR Scheduler(Weighted Round Robin) project in a team of three.
    - Programmed and debugged WRR algorithm into linux kernel on Samsung Artik device, with load-balancing between 8 cores.