Carbon-Aware Serverless System - Project Proposal

Jaykumar Patel
The University of Texas at Austin
patel.jay4802@utexas.edu
jnp2369

Afnan Mir
The University of Texas at Austin
afnanmir@utexas.edu
amm23523

Nidhi Dubagunta
The University of Texas at Austin
nidhi.dubagunta@utexas.edu
nsd632

Abstract

Serverless computing enables developers to build and deploy applications faster without having to worry about provisioning resources and managing infrastructure. Existing serverless computing platforms do not consider carbon footprint when performing dynamic voltage and frequency scaling (DVFS), which scales the frequency that the vCPU runs at and the voltage that is supplied to the vCPU. Other policies, such as resource allocation and keep-alive policies, are also carbon-agnostic. We propose introducing the frequency and voltage at which vCPUs run as a new resource to allocate per invocation. Our system will manage DVFS, along with traditional resource allocation (vCPU, memory), and the keep-alive policy to meet service-level objectives (SLO) while minimizing the carbon footprint. Firstly, we will define how to measure carbon emissions and perform a measurement study, which will involve running a variety of workloads on a variety of policy configurations and acquiring their resource usage, carbon emission, and performance. This will allow us to understand carbon emission during different stages of a serverless function's lifecycle, how carbon emission varies with changes in vCPU and memory allocation, how carbon emission varies with different inputs to the same serverless function, and how carbon emission varies with dynamic voltage and frequency scaling (DVFS). Example workloads include running database queries, data processing tasks, and machine learning inference tasks. Then, we will adapt OpenWhisk, an open source and serverless cloud platform, to incorporate machine learning algorithms to drive a carbon-aware DVFS and resource allocation model, where voltage and frequency to the vCPU will be considered as a resource along with memory and vCPU. We will use this framework to investigate whether machine learning techniques are effective in driving carbon-aware decisions by evaluating our framework's carbon emissions against OpenWhisk's current policies. Further work may include creating a framework for the keep-alive policy and scheduling to further minimize carbon footprint. The ultimate goal of this project is to minimize the carbon footprint while meeting each invocation's SLOs, which are tailored for the individual workloads and determined by metrics such as latency, throughput, total runtime, resulting cost, etc. Though our system should contribute to environmental sustainability, it must also consistently meet the specified operational criteria essential for their intended functions.

1 Timeline

- **Week 1:** Set up OpenWhisk, which is a serverless and open source cloud platform. This will allow us to deploy and invoke serverless functions.
- **Weeks 2 and 3:** Research and implement a measurement tool to determine the carbon footprint of cloud computing. This will be used to measure the carbon footprint of serverless function invocations.

- **Week 4:** Develop and run a measurement study. This will involve running a variety of workloads on a variety of policy configurations and measuring the carbon footprint, performance, and resource usage, such as vCPU and memory, for each invocation.
- **Week 6 and onwards:** The following weeks will be spent on developing a carbon-aware serverless system. The system will involve the following components, in order of priority:
    1. **DVFS and Resource Allocation:** Develop a framework that leverages machine learning to drive carbon-aware decisions regarding DVFS and resource allocation. This entails optimizing memory and vCPU allocation as well as frequency and voltage settings for vCPUs to minimize carbon footprint.
    2. **Keep-Alive Policy:** Develop a keep-alive policy that keeps serverless functions alive for a certain amount of time to reduce the carbon footprint of cold starts.
    3. **Scheduling:** Develop an algorithm that schedules serverless function invocations to minimize the carbon footprint.