CSE 291D: Virtualization

Convert Programs to Serverless Functions

Team Members

- Pratyush Karmakar - A59012917
- Prajwal Yelandur Raghu - A59008845
- Raghasuma Aishwarya Putchakayala - A59002409
- Sruthi Praveen Kumar Geetha - A59012417

Problem Statement / Synopsis

The main objective of the project is to build a web application that executes Python code in a browser environment by compiling it into Web Assembly using serverless functions. The project also aims to compare the performance of the serverless compiler with that of a server-based compiler.

Importance

This project proposes to delegate the server management and maintenance of the compiler to a third-party cloud provider (AWS). Benefitting from the numerous advantages that serverless architecture provides like greater scalability, higher accessibility at a lower cost, over the traditional server-centric architecture, the proposed compiler will allow the users to compile Python scripts without the need to install any other plugins. The proposed compiler also offers faster performance compared to conventional Python interpreters due to the fact that the WASM code (binary executable) is generated for the entire input source program at once which is then executed in a browser, whereas a conventional Python interpreter translates just one statement of the input source program at a time.

Approach

The first step is to explore the AWS services like AWS Lambda and Step Functions, and determine the additional services (if any) required to implement the serverless application.

The compiler takes input as Python code and generates WASM code as output which can be executed in a browser. The compiler supports a subset of Python. It has three stages:

- Parsing - Parses the source code into an abstract syntax tree format
- Type Checking - Performs the static typing of the variables, functions and classes
- Code Generation - Converts the parsed and typed code into WASM code to be executed on a browser

The second step involves determination of the different stages in the workflow for the serverless application by using the above stages of a compiler, implemented using AWS Lambda and Step functions.

The third step consists of testing the serverless application to validate its intended functionality.

The final step involves performing experiments and comparison of the performance between the implemented serverless application and the server-based application by using evaluation metrics such as CPU, memory utilization and speed.

Challenges

Considering the fact that the compiler is a compute-intensive system software with phases from parsing to code generation, porting it to serverless architecture is a challenging task in itself. Below are the various challenges that we might encounter during the project implementation and experimentation:

- To perform the resource profiling of the compiler and compare its resource demands in a server-centric vs serverless environment settings, the compiler software might have to be “splitted” at a more granular level than the phases already identified.
- Getting a handle on the testing and experimentation tools for serverless components might seem demanding.
- Since AWS Lambda, which is the serverless computing platform being used for the project, is relatively new to the team, it calls for extensive exploration in the initial phase. Combined with the usage of the platform for implementation of the compiler and the comprehensive experimentation in the end, it could be a challenge to complete the project within the offered AWS credits.

Appendix

Requirement Specifications

1. Hardware Requirements:
    - 8 GB RAM
    - Quad Core i5 Processor
2. Software Requirements:
    - A web browser like Chrome/Firefox/Edge/Safari for the client/user
    - EC2 instance for the server-based compiler
    - AWS Lambda and Step Functions for the serverless compiler
    - TypeScript for development

Milestones

Week	Progress
4	Choice of Problem Statement and Project Proposal
5	Exploring AWS Lambda and Step Functions
6	Designing a workflow using AWS Step Functions
7	Progress Report
8	Testing and Experiments
9	Comparison of server based and serverless compiler
10	Project Presentation and Report