Principle of Programming Languages: Extra class

MEng. Tran Ngoc Bao Duy, Dr. Nguyen Hua Phung

Jan, 2024

1 Introduction

As concerned in the first chapter, the phases in compilation process can be grouped into two main groups: front-end and back-end. The output of the front-end is intermediate code which can be AST. The output of the back-end is machine code. In the main stream of this course, Java byte code which is used to run on Java Virtual Machine(JVM) is used as the output of the back-end phase. JVM is a stack-based machine and accepts object-oriented paradigm. It is easier to generate code for a stack-based machine with the trade-off on performance.

In this extra class, you are required to generate code for a virtual register-based machine which is based on MIPS (Microprocessor without Interlocked Pipelined Stages) architecture. The virtual machine is used to test your solution is MARS 4.5 which is used in Computer Architecture course.

To complete this requirement, you need to

- Review your knowledge about MIPS which was studied in Computer Architecture course.
- Watch the movies about JVM and Code Generator
- Download the initial code and read it to have some idea to build modularly a code generator for MIPS
- Read chapter "Code Generation" of [1].
- Implement a code generator for MIPS architecture.

2 Tasks

You are required to complete the following tasks and submit them by corresponding deadlines :

1. Extend the initial code such that it can generate JVM code for binary expression with + operator.

2. Redesign the intial code such that it can generate MIPS code for the same inputs of task 1. Implementing the algorithm to assign registers and manage variables is required in this phase.

3. Extend the solution of task 2 such that it can generate MIPS code for the ZCode language.

3 Submission

You are required to submit:

1. a report represents the current architecture of the initial code generator together with 4 files: ZCode.g4, ASTGeneration.py, CodeGenerator.py and CodeGenSuite.py

2. a report represents your design of the MIPS code generator and a zip file named "task2.zip" that contains your code such that after unzip your file, go into task2/src folder and run "python3 run.py test CodeGenSuite" successfully.

3. a final report represents your design of the MIPS code generator and a zip file named "task3.zip" that contains your code such that after unzip your file, go into task3/src folder and run "python3 run.py test CodenGenSuite" successfully. The CodeGenSuite.py must include 100 testcases which can cover all features of ZCode.

The deadline of the submission is provided in the submission task on the website

4 Evaluation

The student must pass all tasks to be consider passing this course. The minimum requirement for each task as follows:

1. Task 1: Submit the report and 6 files on time. The code can be compiled and tested successfully 3 test-cases with + operator.

2. Task 2: Submit the report and the zip file on time. The reports represents the modular design and the algorithm used to assign registers and manage variables. The code can be compiled and tested successfully 3 test-cases with + operator.

3. Task 3: Submit the report and the zip file on time. The reports represents the modular design and the algorithm used to assign registers and manage variables. The code can be compiled and tested successfully at least half of the testcases. The CodeGenSuite.py contains 100 testcases of assignment 4.

You must complete the project (both report and code) by yourself and do not let your work seen by someone else, otherwise, you will be punished by the university rule for plagiarism.

References

[1] Alfred V. Aho, Monica S. Lam, Ravi Sethi, and Jeffrey D. Ullman. *Compilers: Principles, Techniques, and Tools (2nd Edition)*. Addison-Wesley Longman Publishing Co., Inc., USA, 2006.