Homework 2 - SSA Construction for Bril

CSIE 5054 - Advanced Compiler Design
National Taiwan University

Total Points:	8
Release Date:	2024/10/18 08:00
Due Date:	2024/11/08 23:59
TA Email:	llvm@csie.ntu.edu.tw

Contents

1 Announcement 1

2 Preliminaries 1

3 Problem 1 1

4 Plagiarism and Academic Integrity 4

1 Announcement

1. The homework contains only 1 problem.
2. In case you encounter technical issues with Homework 2, kindly consult online resources initially, as most problems are likely related to your local environment. If challenges persist, reach out to our TAs following these guidelines:
   - Title your email [AC-HW2][Summary-Of-Your-Issue]. Please note that we will **NOT** receive emails in other formats as we have applied filters to our email system.
   - Provide detailed information about your computer, including the operating system.
   - Outline the methods you attempted previously, the resources you consulted, the steps you followed, and the results of your efforts.
   - Note that ambiguous requests, such as attaching screenshots without proper descriptions, will not be answered. **Such emailing will lower your priority.**
   - For guidance on how to formulate effective technical inquiries, please refer to How To Ask Questions The Smart Way. This resource can help you structure your questions to get quicker, more precise responses from the TAs.

2 Preliminaries

- Git
- GitHub
- Bril Tool-chain

3 Problem 1

In this homework, you will implement the SSA construction algorithm for the Bril intermediate representation (IR). The goal is to transform a given non-SSA Bril program into its SSA form with considering the optimization of eliminating redundant ϕ-functions. You will work through several modular components, each building upon the previous to achieve the final SSA-form program.

3.1 Task Details

3.1.1 Overview

1. **Set Up the Environment**: Ensure your development environment is ready for Bril development.
2. **Implement Modules**: Complete the provided starter code by implementing the required modules step by step.
3. **Validate SSA Form**: Check that the output program is in valid SSA form.
4. **Test Your Implementation**: Use provided test cases to verify the correctness of your SSA construction.

3.1.2 Steps

Below are the detailed steps to guide you through the homework:

1. **Accept the Assignment and Clone the Repository**

```
git clone --recursive <TBA>
cd <homework-directory>
```

2. **Review the Starter Code**: The repository contains the following structure:

```
homework-directory/
├── src/
│   ├── driver.py
│   ├── bril.py
│   ├── cfg.py
│   ├── dominance.py
│   ├── is_ssa.py
│   └── ssa_construct.py
├── tests/
│   ├── simple.bril
│   └── [additional test cases]
├── bril/
├── install_bril.sh
├── run_test_case.sh
├── student_id.txt
└── README.md
```

3. **Implement the CFG Construction in cfg.py**: Build the Control Flow Graph (CFG) by identifying basic blocks and establishing control flow between them.
4. **Implement Dominator Calculations in dominance.py**: Compute dominators, immediate dominators, and dominance frontiers for each basic block.
5. **Insert ϕ-Functions in ssa_construct.py**: Use the dominance frontiers to identify the correct placement of ϕ-functions for each variable and insert them into the CFG.
6. **Implement Variable Renaming in ssa_construct.py**: Rename variables to ensure each one is assigned exactly once, adhering to SSA form.

3.2 Testing Guideline

- To generate the SSA form of your program:

```
bril2json < ./tests/[your_test_program].bril | python3 ./src/driver.py |
    bril2txt > output.bril
```

- To check if the generated program is in SSA form:

```
bril2json < ./output.bril | python3 src/is_ssa.py
```

  If output is "Not SSA", your SSA transformation is incorrect, and you should debug your implementation.

- To compare the execution output of the original and transformed programs:

```
# Original program
bril2json < ./tests/[your_test_program].bril | brili [arguments] > original.out

# Transformed SSA program
bril2json < ./output.bril | brili [arguments] > transformed.out

# Compare outputs
diff original.out transformed.out
```

  If there is any difference between original.out and transformed.out, your SSA transformation is incorrect, and you should debug your implementation.

3.3 Submission Instructions

1. **Add your student ID to student_id.txt**: Open student_id.txt and replace the placeholder with your actual student ID.
2. **Ensure all your code changes are within the src/ directory**: Only modify files inside the src/ directory. **DO NOT** alter other parts of the repository unless explicitly instructed.
3. **Commit and Push Your Changes**

```
git add src/ student_id.txt
git commit -m "Completed Homework 2"
git push origin main
```

4. **Check the GitHub Actions Workflow**
5. **Confirm Grading Output**

3.4 Do and Don't

- **You are allowed to modify any part of the starter code within the src/ directory, except for is_ssa.py, to suit your approach.** While the current structure serves as a guideline, ensuring the driver script functions properly is key for grading.
- **Make sure you have a solid understanding of the algorithm before starting your implementation.**
- Ensure your student ID is correctly entered in the student_id.txt file before submission.
- **DO NOT** modify the src/is_ssa.py file or anything outside the src/ directory except student_id.txt. Any such changes will be considered cheating.
- Please note that we will be able to see through the GitHub Classroom backend if you have made changes to files that should not be modified.

4 Plagiarism and Academic Integrity

It is important to adhere to the university's academic integrity policy while completing this assignment. Please keep the following in mind:

1. **Do Not Share Your Code**: All submissions must be your own work. Sharing your code with others or obtaining code from others, including online resources, is considered plagiarism and will be treated as academic misconduct.
2. **Use of External Resources**: You are encouraged to textbooks, lecture notes, and official documentation to assist your understanding. However, directly copying code or solutions from external sources (e.g., GitHub, StackOverflow, or similar) without attribution is not allowed.
3. **Collaboration Guidelines**: You may discuss general concepts and strategies with classmates, but all coding and detailed design decisions must be completed **independently**.
4. **Consequences of Plagiarism**: Plagiarism, cheating, or any form of academic dishonesty will result in a zero on the assignment, and may lead to further disciplinary action as per university regulations.