Project Report of MxStar Compiler

(Course Project of Compiler)

Weixin Deng (邓伟信)
ACM Class, Shanghai Jiao Tong University

1 Introduction

This project is a compiler of MxStar[1] language(a combination of C and Java), implemented in Java. The GitHub repository of my project is: https://github.com/dengwxn/Mx-Compiler.

2 Implementation

Source Language	MxStar
Target Language	x86-64 NASM
Lexer/Parser Generator	antlr4

2.1 IR Generation

Logical Expressions I do not use `set` instruction when dealing with logical expressions. Instead, I maintain two blocks `logicTrue` and `logicFalse` as the logic exit blocks. In `assign` statements, I add `mov 0/1` accordingly. Otherwise in `if` statements, `thenStmt` generate IR in `logicTrue` and `elseStmt` in `logicFalse`.

2.2 Optimization

- Global constant propagation.
- Global copy propagation.
- Dead code elimination using neededness analysis.

  - Remove unneeded instructions.
  - Remove empty blocks.
  - Remove unreachable blocks.
  - Remove some useless loops using a conservative strategy.
  - Compress `jump` when jumping to a block consisting of a `jump` instruction.
- Function inlining.
- Global variable loading.
  - Load them in the beginning of a function and store back in the end.
  - Extra storing and loading might be needed when having `call`.
- Value numbering on extended basic blocks.
- Strength reduction.
- Chordal graph coloring.
  - Although I do not convert my program to SSA, the result of register allocation is not bad.

3 Discussions

Debug I use gdb to debug the assembly code. The complete script to support gdb is `nasm -felf64 prog.asm -gdwarf && gcc prog.o -o prog -g -no-pie`.

Object Oriented Programming My implementation is far away from OOP style. I think using OOP will make the whole project clearer but also make it harder to code.

4 Appendix

References

[1] MxStar Language Manual.
https://acm.sjtu.edu.cn/w/images/3/30/M_language_manual.pdf.

[2] CMU 15-411 Compiler Design.
http://www.cs.cmu.edu/~janh/courses/411/18/schedule.html.

[3] Stanford CS143 Compilers.
http://web.stanford.edu/class/cs143/index2018.html.

[4] Lianmin Zheng's Compiler project.
https://github.com/merrymercy/compiler2017/blob/master/doc/report.pdf.

[5] Zhekai Zhang's Compiler project.
https://github.com/sxtyzhangzk/MxCompiler/blob/master/slide.pdf.

[6] 青木峰郎. 自制编译器, 人民邮电出版社, 2016.