CSE 501: Implementation of Programming Languages
Programming Assignment 1

Haichen Shen

1 Description: CFG Construction and Dominators

The goal is to construct the control flow graph for each method and then find the immediat dominator for each basic block.

2 Implementation

Inside this project, I use the algorithm created by Cooper, Harvey and Kennedy to find the immediate dominators. The main idea of this method is finding the closest common immdiate dominator of all predecessors of one node.

Main flow of my program is:

1. Read the whole start file, parse the routines inside this file and create a class *Routine* for each routine.
2. For every routine, find the basic blocks inside the method, create a class *BasicBlock* for each basic block, and generate the CFG.
3. Now iterate each routine and run the idom algorithm:
   - 3a. Generate postorder sequence of all block.
   - 3b. According to inverse postorder, iteratively update the idom for each block.

The project is implemented in Java.

3 Measurements

I measured the time consumption in two parts respectively: (i) finding the basic and building CFG, (ii) finding dominators. I ran 10 times for a program and use the average time as the result. Firstly, I ran the tests on all example programs. The result is shown in Table 1.

I think the reason that it take quite a long time to build the control flow graph is because (i) firstly it takes some time to go through the all instructions the find out all basic blocks and go through the blocks again to build the control flow graph; (ii) I also copy the instructions to each block and it could slow down the program.

Besides I notice that the time in finding dominatros are related to the max basic blocks in a routine. But it is uncertain what kind of relationship they are.

Program	Build the CFG(μs)	Find dominators(μs)	Total basic block	Max basic block in a routine
points	2538.36	403.39	8	4
mmm	4247.37	942.76	44	40
cd	2437.39	363.11	5	4
hanoifibfac	2836.37	490.58	15	4
rime	2746.81	500.03	13	13
rational	3545.57	569.73	19	9
truct	3159.37	421.92	7	7
ieve	2794.85	524.84	16	16
link	2540.44	411.33	9	4
oop	2755.69	584.24	19	19
lass	2486.10	365.65	5	3
sort	2904.28	559.56	18	18
regslarge	14727.42	354.22	4	4

Table 1: Time consumption in building CFG and finding dominators

Therefore I write a small program which generatee loop program to figure out the relationship between the block numbers and time consumption in finding idom. The name of prgroam means how many layers of loop inside. And every test program has only the main function. The result of the my testing programs shows in Table 2.

Program	Find dominators(μs)	Basic blocks
loop20	1178.58	61
loop40	1971.86	121
loop80	3501.38	241
loop100	4287.60	301
loop200	8536.69	601
loop400	15903.11	1201
loop800	32763.57	2401

Table 2: Time consumption test

In the Figure 1, it is apparent that the time grows linearly along with the number of basic blocks which means that the complexity of algorithm should be $O(n)$. I haven't tried other programs. Since the loop program is quite simple and doesn't have a complex CFG, probably it's only for the simple program that the time consumption is linear to number of blocks.

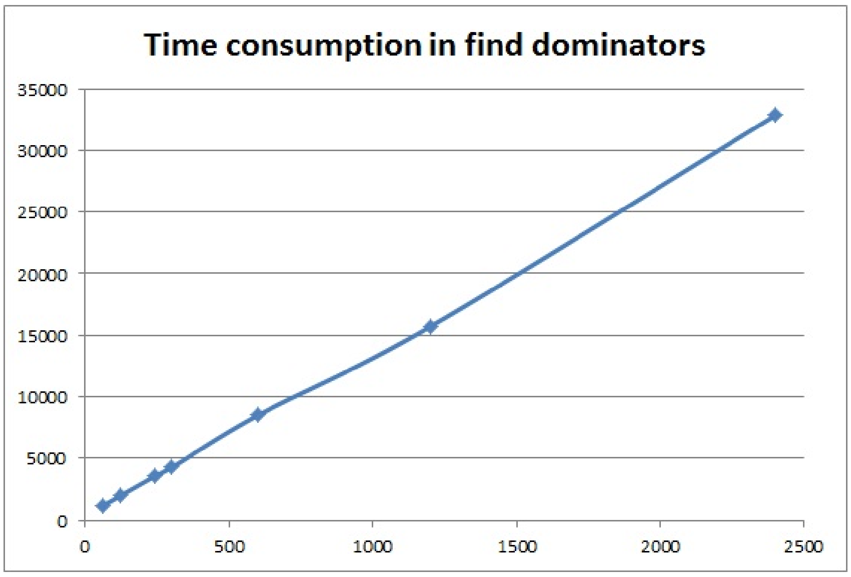

Fig. 1: Relationship between block number and time consumption. X-axis: the number of basic blocks, Y-axis: the time consumption (μs)