CS204 - Mini Project 2: Branch Prediction

Harjas Kaur: 2022eeb1171

Manas Deorah

Ragini Gupta

# Details of Code:

Compile with simple gcc main.c , paste the trace in trace.txt. for files less than 20,000 lines

For very large inputs,

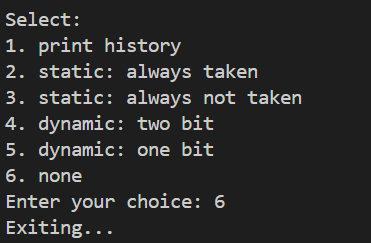
gcc main.c -o program -mcmodel=medium

./program

Made on Ubuntu (WSL2).

Other explanations are given in the comments.

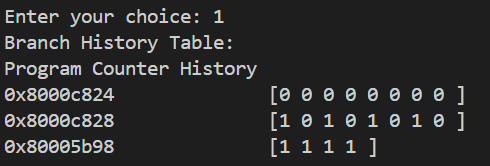
Interface:



# Test Cases:

## Case 1:

Used a small code snippet to check minor details of the code. Code is given in the Appendix(1).



Since the BTB tables are very large I am only giving the accuracy here:

For these small codes the accuracies seem correct.

(this one was 100% entirely by chance, since the code is very random)



## Case 2: Bubble Sort code

Used first 10425 lines, link given in appendix(2).

## Case 3: Fac\_test\_lab code

Used first 10523 lines, link given in appendix(3).

## Case 4: q\_sort code

Used first 20478 lines, link given in appendix(4).

# Observations:

## Static predictors:

Both kinds of predictors generally have (relatively) poor accuracy in real application codes.

## Dynamic Predictors

For larger codes, as I have only tested for a fraction of the whole code, two bit branch predictor’s accuracy is slightly worse as its only advantage in increasing accuracy is at loop termination conditions. As my input is a very small subset of that code, it might be encountering very few loop termination conditions, giving worse accuracy overall . For giving large inputs the slightest errors in copy pasting is causing segmentation faults, so I limited input to 20 thousand lines which is a fraction of the whole code (approx. 4,00,000 lines).

## Improvements

The program has very large time complexity for dynamic predictors which is noticeable when we give inputs above 80,000 lines.

The use of arrays was not a wise decision to parse the files as it is very large so compiler flags have been used to address that issue. Using the “-mcmodel=medium” flag gave the program a memory of 2GB to work on but it still was insufficient. This can only be improved by using a better data structure.

The maximum number of instructions this can process might also be affected due to my extensive use of arrays. I need to test it further.

# Appendix

1. Test code used for debugging:

core 0: 0x8000c81c (0x00150513) addi a0, a0, 1

core 0: 0x8000c820 (0xfff5c703) lbu a4, -1(a1)

core 0: 0x8000c824 (0x00078863) beqz a5, pc + 16

core 0: 0x8000c828 (0xfee786e3) beq a5, a4, pc - 20

core 0: 0x8000c814 (0x00054783) lbu a5, 0(a0)

core 0: 0x8000c818 (0x00158593) addi a1, a1, 1

core 0: 0x8000c81c (0x00150513) addi a0, a0, 1

core 0: 0x8000c820 (0xfff5c703) lbu a4, -1(a1)

core 0: 0x8000c824 (0x00078863) beqz a5, pc + 16

core 0: 0x8000c828 (0xfee786e3) beq a5, a4, pc - 20

core 0: 0x8000c82c (0x40e78533) sub a0, a5, a4

core 0: 0x8000c830 (0x00008067) ret

core 0: 0x80005b98 (0x02051663) bnez a0, pc + 44

core 0: 0x80005bc4 (0x003c0c13) addi s8, s8, 3

core 0: 0x80005bc8 (0x41fc5793) srai a5, s8, 31

core 0: 0x80005bcc (0x0037f793) andi a5, a5, 3

core 0: 0x80005bd0 (0x018787b3) add a5, a5, s8

core 0: 0x80005bd4 (0x004b2703) lw a4, 4(s6)

core 0: 0x8000c81c (0x00150513) addi a0, a0, 1

core 0: 0x8000c820 (0xfff5c703) lbu a4, -1(a1)

core 0: 0x8000c824 (0x00078863) beqz a5, pc + 16

core 0: 0x8000c828 (0xfee786e3) beq a5, a4, pc - 20

core 0: 0x8000c814 (0x00054783) lbu a5, 0(a0)

core 0: 0x8000c818 (0x00158593) addi a1, a1, 1

core 0: 0x8000c81c (0x00150513) addi a0, a0, 1

core 0: 0x8000c820 (0xfff5c703) lbu a4, -1(a1)

core 0: 0x8000c824 (0x00078863) beqz a5, pc + 16

core 0: 0x8000c828 (0xfee786e3) beq a5, a4, pc - 20

core 0: 0x8000c82c (0x40e78533) sub a0, a5, a4

core 0: 0x8000c830 (0x00008067) ret

core 0: 0x80005b98 (0x02051663) bnez a0, pc + 44

core 0: 0x80005bc4 (0x003c0c13) addi s8, s8, 3

core 0: 0x80005bc8 (0x41fc5793) srai a5, s8, 31

core 0: 0x80005bcc (0x0037f793) andi a5, a5, 3

core 0: 0x80005bd0 (0x018787b3) add a5, a5, s8

core 0: 0x80005bd4 (0x004b2703) lw a4, 4(s6)

core 0: 0x8000c81c (0x00150513) addi a0, a0, 1

core 0: 0x8000c820 (0xfff5c703) lbu a4, -1(a1)

core 0: 0x8000c824 (0x00078863) beqz a5, pc + 16

core 0: 0x8000c828 (0xfee786e3) beq a5, a4, pc - 20

core 0: 0x8000c814 (0x00054783) lbu a5, 0(a0)

core 0: 0x8000c818 (0x00158593) addi a1, a1, 1

core 0: 0x8000c81c (0x00150513) addi a0, a0, 1

core 0: 0x8000c820 (0xfff5c703) lbu a4, -1(a1)

core 0: 0x8000c824 (0x00078863) beqz a5, pc + 16

core 0: 0x8000c828 (0xfee786e3) beq a5, a4, pc - 20

core 0: 0x8000c82c (0x40e78533) sub a0, a5, a4

core 0: 0x8000c830 (0x00008067) ret

core 0: 0x80005b98 (0x02051663) bnez a0, pc + 44

core 0: 0x80005bc4 (0x003c0c13) addi s8, s8, 3

core 0: 0x80005bc8 (0x41fc5793) srai a5, s8, 31

core 0: 0x80005bcc (0x0037f793) andi a5, a5, 3

core 0: 0x80005bd0 (0x018787b3) add a5, a5, s8

core 0: 0x80005bd4 (0x004b2703) lw a4, 4(s6)

core 0: 0x8000c81c (0x00150513) addi a0, a0, 1

core 0: 0x8000c820 (0xfff5c703) lbu a4, -1(a1)

core 0: 0x8000c824 (0x00078863) beqz a5, pc + 16

core 0: 0x8000c828 (0xfee786e3) beq a5, a4, pc - 20

core 0: 0x8000c814 (0x00054783) lbu a5, 0(a0)

core 0: 0x8000c818 (0x00158593) addi a1, a1, 1

core 0: 0x8000c81c (0x00150513) addi a0, a0, 1

core 0: 0x8000c820 (0xfff5c703) lbu a4, -1(a1)

core 0: 0x8000c824 (0x00078863) beqz a5, pc + 16

core 0: 0x8000c828 (0xfee786e3) beq a5, a4, pc - 20

core 0: 0x8000c82c (0x40e78533) sub a0, a5, a4

core 0: 0x8000c830 (0x00008067) ret

core 0: 0x80005b98 (0x02051663) bnez a0, pc + 44

core 0: 0x80005bc4 (0x003c0c13) addi s8, s8, 3

core 0: 0x80005bc8 (0x41fc5793) srai a5, s8, 31

core 0: 0x80005bcc (0x0037f793) andi a5, a5, 3

core 0: 0x80005bd0 (0x018787b3) add a5, a5, s8

core 0: 0x80005bd4 (0x004b2703) lw a4, 4(s6

1. Used first instructions of the codes given in the links
2. Code 1: https://drive.google.com/file/d/1khHi-K3gddnvoIX1U-DSGQjsnXlZRxMk/view?usp=drive\_link
3. Code 2: https://drive.google.com/file/d/1SMwX-CzZO9R2jNIOXhLJiTtxqetmfUkA/view?usp=drive\_link
4. Code 3: https://drive.google.com/file/d/1Sd0b\_s7\_OlByMq\_YNb7qH8qBKxnBik1c/view?usp=drive\_link