A6: Evaluering af ydeevne – teori

Computer Systems 2021
Department of Computer Science
University of Copenhagen

Finn Schiermer Andersen/Michael Kirkedal Thomsen

Due: Søndag, 9. Oktober, 16:00
Version 1

Dette er den fjerde afleveringsopgave i Computersystemer og den anden (og sidste) som dækker Maskinarkitektur. Som tidligere, opfordrer vi til parprogrammering og anbefaler derfor at lave grupper at 2 til 3 studerende. Grupper må ikke være større end 3 studerende og vi advarer mod at arbejde alene.

Afleveringer vil blive bedømt med op til 4 point. Du skal opnå mindst halvdelen af de mulige point over kurset, samt mindst 3 point i hvert emne, for at blive indstillet til eksamen. Denne aflevering hører under Maskinarkitektur. Du kan finde yderligere detaljer under siden "Course description" på Absalon. Det er *ikke* muligt at genaflevere afleveringer.

1 Pipeline

Denne opgave tager udgangspunkt i beskrivelsen af en 2-vejs superskalar in-order mikroarkitektur, som præsenteret i noten om afviklingsplot her:

https://x86prime.github.io/afviklingsplot/superscalar/

Opgaven går ud på at vurdere køretid for en lille del af et program.

Programmet summerer elementerne i en nul-termineret vektor. Den indre løkke ser således ud i x86prime:

```
.L3:
    addq $8, %rdi
    addq %rdx, %rax
    movq (%rdi), %rdx
    cbne $0, %rdx, .L3
```

1.1 Spørgsmål 1

Tegn et afviklingsplot som viser 2 gennemløb af løkken på en 2-vejs superskalar som den er beskrevet i noten om afviklingsplot. Antag at såvel instruktions- som data-cache har en tilgangstid på en enkelt clock-periode. Vi ser bort fra cache miss.

1.2 Spørgsmål 2

Hvad er CPI for afviklingen af den indre løkke på den angivne maskine i spørgsmål 1?

Det er muligt at bruge en anden maskine i stedet. Denne anden maskine er også en 2-vejs superskalar, men den har en 25 procent højere clock-frekvens. Til gengæld er tilgangstiden til såvel instruktions- som data-cache på 2 clock perioder.

1.3 Spørgsmål 3

Tegn et afviklingsplot som viser 2 gennemløb af løkken på denne nye maskine. Se igen bort fra cache miss.

1.4 Spørgsmål 4

Hvad er CPI for afviklingen af den indre løkke på den angivne maskine i spørgsmål 3?

1.5 Spørgsmål 5

Hvilken af de to maskiner afvikler den indre løkke hurtigst?

2 Assembler

Nedenfor er givet en funktion i x86prime maskinsprog.

```
.p:
.LFB0:
    subq $8, %rsp
    movq %r11, (%rsp)
    movq (%rdi), %rax
    movq $0, %edx
.L2:
    cbe $0, %rax, .L4
    addq %rax, %rdx
    addq $8, %rdi
    movq (%rdi), %rax
    jmp .L2
.L4:
    movq %rdx, %rax
    movq (%rsp), %r11
    addq $8, %rsp
    ret %r11
```

Besvar følgende spørgsmål:

2.1. Angiv for hver register om det bruges til at holde en pointer eller en long

2.2. Udpeg de instruktioner som implementerer kald og retur

2.3. Beskriv programforløbet - identificer betinget kode, løkker og eventuelle kald af andre funktioner

2.4. Find og beskriv de betingelser som bestemmer eventuel betinget kode eller løkker

2.5. Opstil en C kildetekst der svarer til funktionen.

3 Cache

Et tensor produkt af to vektorer udregnes hvert element af vektor A ganges som en skalar til en vektor B. Resultatet er altså en matrice at størrelse to vektorer.

Se mere her: `https://en.wikipedia.org/wiki/Tensor_product`

Følgende C-lignende program kan udregne dette:

```
void tensorproduct(int A[n], int B[m], int C[n][m])
{
    for (int i = 0; i < n; i++) {
        for (int j = 0; k < m; k++) {
            C[i][j] = A[i] * B[j];
                }
            }
        }
    }
}
```

Beskriv lokaliteten af dette program, dvs.

- Hvad er stride for programmet?
- Har det rummelig (spacial) og temporal lokalitet.

- Kan programmet opnå en bedre lokalitet (både rummelig og temporal) og hvordan?

4 Bedømmelse og rapportering

Rapporten skal berøre problemstillinger som er rejst under opgaven, inklusiv de overvejelser som opgaveteksten lægger op til. Beskriv alle ikke-trivielle dele af jeres løsning, samt evt. tvetydige formuleringer, som I kan have fundet i opgaveteksten.

Dertil skal der afleveres `group.txt` som indeholder en ASCII/UTF8 formateret liste KU-id'er fra alle medlemmer i gruppen; et id pr. line, ved brug af følgende tegnsæt:

$$\{\texttt{0x0A}\} \cup \{\texttt{0x30},\texttt{0x31},\ldots,\texttt{0x39}\} \cup \{\texttt{0x61},\texttt{0x62},\ldots,\texttt{0x7A}\}$$