Problem-1

average of Perl bench and Sjeng

$$\text{Loads} = \frac{25+19}{2} = 22\%$$

$$\text{stores} = \frac{14+7}{2} = 10.5\%$$

$$\text{Branches} = \frac{15+15}{2} = 15\%$$

$$\text{Jumps} = \frac{7+3}{2} = 5\%$$

$$\text{ALU} = \frac{39+56}{2} = 47.5\%$$

~~CPI = 0.22 X 3.5 + 0.105 X 2.8 + 0.15 X (~~

CPI when Branch is taken

$$= 0.22 \times 3.5 + 0.105 \times 2.8 + 0.15 \times 4 + 0.05 \times 2.4 + 0.475 \times 1$$

$$= 0.77 + 0.294 + 0.6 + 0.12 + 0.475$$

$$\text{CPI} = 2.259$$

CPI when Branch is not taken

$$= 0.77 + 0.294 + 0.15 \times 2 + 0.12 + 0.475$$

$$= 1.659 + 0.3 =$$

$$\text{CPI} = 1.959$$

CPI with avg. of Branch = $\frac{4+2}{2} = 3$

CPI = 1.659 + 0.15 X 3

CPI = 2.109

Problem-2

HPC4473

Enhanced mode is used 100% of the 50% time it takes in enhanced mode.

Going in reverse order

Time without enhancement = 50% unaffected + 50% $\cdot \left(\frac{1}{\frac{1}{10}}\right)$

$= 550\%$

$$\text{speed up} = \frac{550\%}{100\%} = \underline{5.5}$$

using Amdahl's law to calculate percentage of enhancement

$$5.5 = \frac{1}{1-X+\frac{X}{10}}$$

$$5.5 = \frac{1}{1-0.9X}$$

$$1-0.9X = \frac{1}{5.5}$$

$$1-0.9X = 0.18$$

$$0.9X = 1-0.18$$

$$X = 0.909 \approx 0.91 \text{ or } 91\%$$

Problem-3

(i) when run in isolation: using Amdahl's law

$$\text{speed up} = \frac{1}{0.6 + \frac{0.4}{2}} = 1.25$$

(ii) $$\text{speed up} = \frac{1}{0.01 + \frac{0.99}{2}} = 1.98$$

(iii) For overall system speed up

first application uses 80% of resources

40% of this 80% is enhanced

time taken = $0.8\left(\frac{0.4}{2} + 0.6\right) = 0.8 \times 0.8$

20% of the resources are unenhanced

⇒ using Amdahl's law

$$\text{speed up} = \frac{1}{0.2 + 0.8 \times 0.8}$$

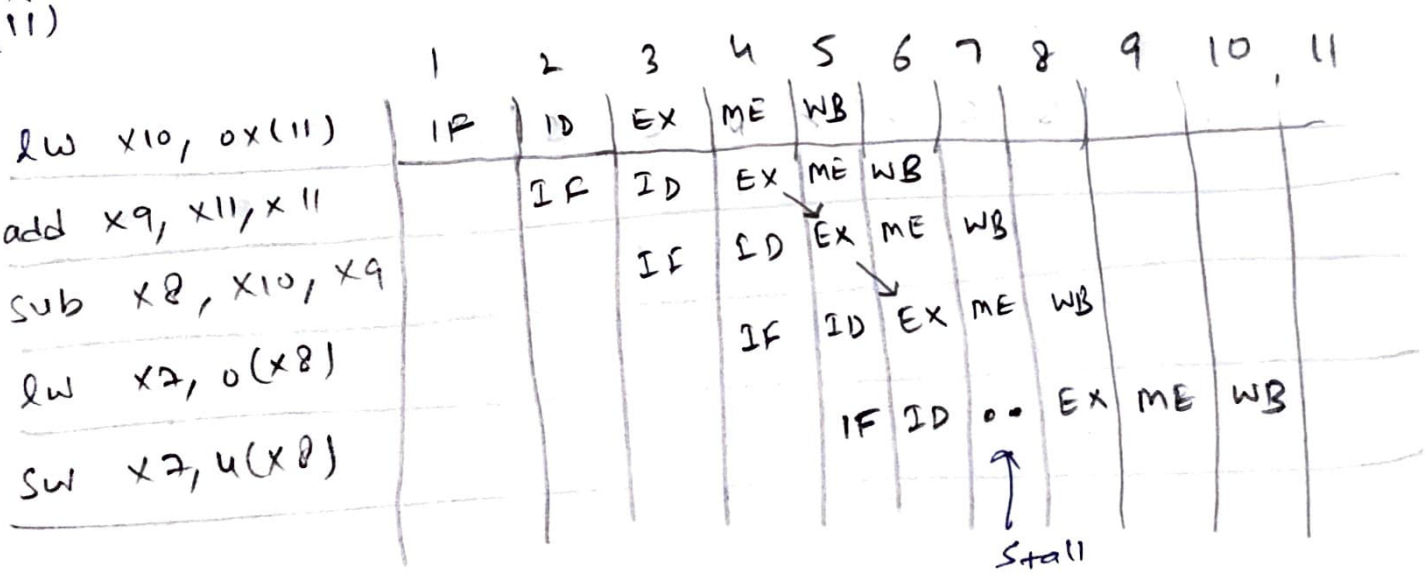
$$= \frac{1}{0.84} = 1.19$$

Problem 4

hpc4473

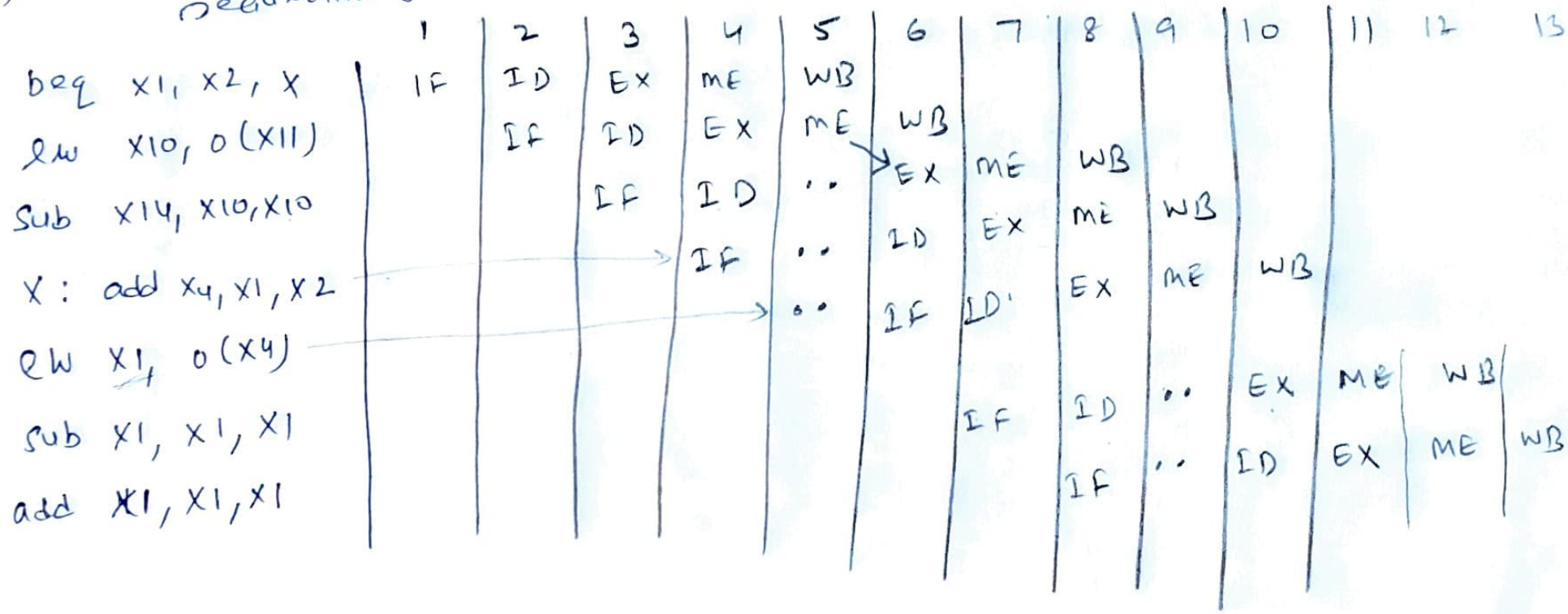
(A) (i) Pipelining improves the throughput by a factor equal to no. of pipeline stages for large numbe of instructions

(ii)



(iii) SW instruction will get its data for register X7 from EX/MEM register.

(B) Given that Branch is not taken so all the instructions will execute sequentially.



Problem-5

(i) More data elements + few blocks:

more data elements mean high spatial locality because block size is big. It can accomodate more sequential data.

Con: This arrangement does not do well when data requirement is heterogeneous.

example: This setup is good for array accesses.

less data elements + more blocks:

Con: It generates more compulsory misses as block size is small

Pro: It can accommodate more heterogeneous data with unique indexes.

example: Data from database.

(ii) word = 8 bits = 1 byte

Block = 16 bytes = 16 words

offset needed = $\log_2 16 = 4$

tag = 11 bits

$\Rightarrow$ index = 20 − 11 − 4 = 5 bits

blocks = 2^5 = 32 blocks

(iii) 16 way set associative

⇒ 16 blocks in a set

addressing : half word

Block size : 16 words : 32 half words

offset = $\log_2 32$ = 5 bits

word length = 64 bits = 8 bytes

Block size = $2^3 \cdot 2^4 = 2^7$ bytes

memory size = 2 MB = 2^{21} bytes

Blocks = $\frac{2^{21}}{2^7} = 2^{14}$ blocks

sets = $\frac{2^{14}}{16} = \frac{2^{14}}{2^4} = 2^{10}$ sets

10 bits are needed to index the sets.

tag = 64 − 10 − 5 = 49 bits

⇒ Tag = 49 bits, offset = 5 bits index = 10 bits