Lab 3 report

Ex3 - Determine the number of instructions executed per cycles (IPC) and MFLOPS. Comment on how they change with increasing number of threads.

The results from the testing can be found in Table 1 below. Multiple runs with the same number of threads were carried out. The run that used the most of the CPU is shown in the table.

The IPC were read directly from the stats gained of the *perf* command.

MFLOPS were calculated as 2*(Matrix size)^3 / seconds. According to the formula (1) below every cell goes through 3 for-loops and then do 2 arithmetic operations (multiplication and addition). So, if we have a 1024x1024 matrix the floiting-point operations would be $2*1024^3 = 2.147$ GFLOP. To get operations per second we divide by the running time. The results are shown in Table 1.

```
for (i = 0; i < size; i++)
    for (j = 0; j < size; j++)                                  (1)
        for (k = 0; k < size; k++)
            result[i][j] += a[i][k] * b[k][j];
```

Table 1

Matrix size:	1024					
Nr of threads:	1	2	4	8	16	32
TowerPC (i7)						
Seconds	8.88	4.56	2.40	1.94	2.21	2.20
CPUs used	1	1.99	3.95	7.70	7.62	7.70
IPC	1.90	1.89	1.89	1.19	1.06	1.05
MFLOPS	241.8	470.9	894.8	1107	971.7	976.1
All-in-one (i5)						
Seconds	10.06	5.34	3.03	3.24	3.25	3.21
IPC	1.92	1.88	2.05	1.92	1.92	1.91
MFLOPS	213.4	402.1	708.7	662.8	660.7	669.0

For the i5 machine the IPC stays the same the whole time. The MFLOPS relates to the running time, which goes down until 4 threads and then stagnates.

For the i7 machine the IPC stays around 1.9 up to 4 threads and then goes down to around 1.1 for more threads. The running time also hits the lowest mark at 8 threads and then stagnates. If all cores were completely utilized for all tests the execution time for 8 threads should be 1/8 of the same for 1 thread and the IPC should stay the same. However, it doesn’t.

From this we can derive that the best utilization for this problem would be to use somewhere between 4 and 8 cores, let’s say 6. Because the i5 only have 4 cores the cores always have enough to do and the IPC stays the same. For the i7, which has 8 cores, there’s not enough work to keep them occupied the whole time so the IPC goes down a bit. Somewhere in the parallelization there is probably some dependencies. The cores have to wait for each other.

The MFLOPS has it’s peak where the cores are best utilized (4 cores for i5 and 8 cores for i7) and the i7 has a higher peak, which is nothing but logical.

E4 - Determine the execution time and derive the speedup of the program

The execution time is shown in Table 1 and are discussed in Ex1.

Speedup: $S_p(n) = \frac{T_{seq}(n)}{T_p(n)}$ Efficiency speedup: $E_p(n) = \frac{T_{seq}(n)}{p \, x \, T_p(n)}$

i7: $S_p = 8.88/1.94 = 4.58$ $E_p = 8.88/(1.94*8) = 0.57$

i5: $S_p = 10.06/3.03 = 3.32$ $E_p = 10.06/(3.03*4) = 0.83$

So in total speedup the i7 is better but for efficiency the i5 is better. This is because of the same reasoning as above. The cores in i5 is better utilized than those in the i7.

E5 - Devise two methods to determine the sequential fraction and the maximum speedup achievable under Amdahl’s Law

Method 1: derive from speedup using Amdahl’s Law

$$S_p(n) = \frac{T_{seq}(n)}{f \, x \, T_{seq}(n) + \frac{1-f}{p} T_{seq}(n)} = \frac{1}{f + \frac{1-f}{p}} = \frac{p}{f(p-1)+1} \rightarrow f = \frac{p - S_p}{s_p(p-1)}$$

i7: f = (8-4.58) / 4.58*(8-1) = 0.1067

i5: f = (4-3.32) / 3.32*(4-1) = 0.0683

They should be the same but the difference may be from inconsistencies in measurements. F is only theoretical in these cases as well.

Method 2: derive from execution time

Execution time = seq + parallel/cores

i7: seq + par/1 = 8.88s

seq + par/8 = 1.94s

→ par = 7.93s, seq = 0.95s

seq time / total time = 0.95/8.88 = 0.107 = f

i5: 0.69s / 10.06s = 0.069 = f

(Method 3: derive from execution time using Amdahl's law)

$$S_p(n)=\frac{T_{seq}(n)}{T_p}=\frac{1}{f+\frac{1-f}{p}} \rightarrow T_p=f*T_{seq}(n)+\frac{(1-f)*T_{seq}(n)}{p} \rightarrow f=\frac{p*T_p-T_{seq}}{T_{seq}*(p-1)}$$

i7: f = (8*1,94-8,88) / (8,88*(8-1)) = 0.1068

i5: f = (4*3,03-10,06) / (10,06*(4-1)) = 0,0682