Sreepathy Jayanand

CED17I038

High Performance Computing Lab-1

Analysis of parallelization factor

The arrays have a size of n. In this case 1e8

The number of threads = num\_of\_threads. In this case 1 - 24

**Strategy:**

Each thread has a “work” of n/num\_of\_threads

Thread1 : 0 to n/number of threads

Thread2 : n/number of threads to 2 \* n/number of threads

And so on….

Table for vector addition:

| Number of Threads | System Time taken for execution |
| --- | --- |
| 1 | 0m1.428s |
| 2 | 0m1.085s |
| 4 | 0m1.119s |
| 6 | 0m1.602s |
| 8 | 0m1.707s |
| 10 | 0m1.769s |
| 12 | 0m1.768s |
| 14 | 0m1.733s |
| 16 | 0m1.765s |
| 20 | 0m1.765s |
| 24 | 0m1.752s |

The most efficient thread size for this program is 2

According to Amdahl’s law :

Speed up = 1/((1-p) + p/N)

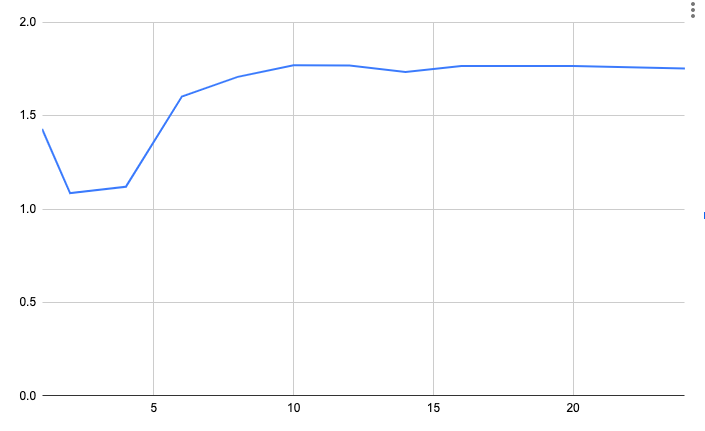
Or

Parallelization factor = (N - N/S.U)/(N - 1)

So here N = 2;

Speed Up = 1.428/1.085 = 1.316

PF = (2 - 2/1.316)/1 = 0.48



Exec time vs no of threads

Y axis - Exec time

X axis - No of Threads

Table for vector multiplication:

| Number of Threads | System Time taken for execution |
| --- | --- |
| 1 | 0m1.303s |
| 2 | 0m1.193s |
| 4 | 0m1.327s |
| 6 | 0m1.757s |
| 8 | 0m1.678s |
| 10 | 0m1.794s |
| 12 | 0m1.791s |
| 14 | 0m1.772s |
| 16 | 0m1.765s |
| 20 | 0m1.765s |
| 24 | 0m1.886s |

The most efficient thread size for this program is 2

According to Amdahl’s law :

Speed up = 1/((1-p) + p/N)

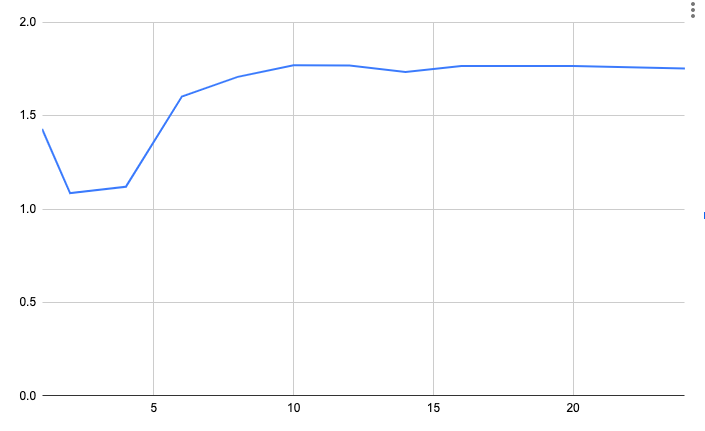
Or

Parallelization factor = (N - N/S.U)/(N - 1)

So here N = 2;

Speed Up = 1.303/1.193 = 1.09

PF = (2 - 2/1.09)/1 = 0.16



Exec time vs no of threads

Y axis - Exec time

X axis - No of Threads