Table 2 - Matrix multiplication compute time time for increasing matrix size

(Serial & Parallel with MPI). **Note: You may reduce the size of the matrices to**

**suit your local computing resources**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Number of nodes (Specify 1 if only using your local computer): 1  Number of CPU cores or logical processes: 2  Number of threads used for POSIX/OMP: 2 | | | | | |
| Matrix size | 500x500 | 1000x1000 | 2000x2000 | 3000x3000 | 4000x4000 |
| Serial time, Ts (s) | 0.49 | 3.91 | 33.14 | 137.18 | 333.25 |
| Parallel time, Tp -  POSIX/OMP (s) | 0.26 | 2.13 | 18.18 | 69.81 | 175.86 |
| MPI  communication  time (s) | 0.00 | 0.00 | 0.92 | 0.20 | 0.21 |
| Speed Up (Ts/Tp) | 1.88 | 1.83 | 1.82 | 1.97 | 1.90 |

The theoretical speed up is approximately 1.818.

To calculate the theoretical speedup when using 2 processes compared to the serial version of your matrix multiplication code, we can use Amdahl's Law. This law states that the speedup of a task using multiple processors is limited by the sequential portion of the task.

Amdahl's Law Formula:

S = 1 / ( (1 - P) + P / N)

Assume that about 90% of the work can be parallelized, then P = 0.9.

S= 1 / ( (1 - 0.9) + 0.9 / 2) ≈ 1.818

The actual speedup values for POSIX/OMP and MPI are generally close to the theoretical speedup (1.818) for smaller matrices (500x500 and 1000x1000). As matrix sizes increase, the actual speedup slightly decreases, especially for the 4000x4000 matrix. The MPI communication time affects the overall speedup and is noticeable in larger matrix sizes. The added communication time leads to a reduced speedup for MPI implementations when compared to POSIX/OMP implementations which do not have communication overhead. For larger matrices (like 4000x4000 and above), the sequential portion of the matrix multiplication especially initialization, cleanup and communication starts to dominate which reduces the potential speedup achievable through parallelization.









