Parallelization Strategy For MPI

To optimize the performance of MPI, it is crucial to minimize communication in parallel programs. This involves reducing data transfer among processes and optimizing communication patterns to minimize global synchronization while promoting local communication.

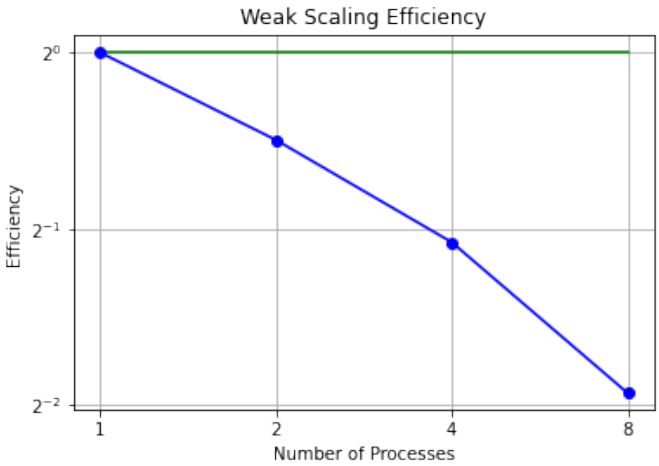
Our primary parallel strategy focuses on reducing unnecessary data transfer through techniques such as **data partition and halo exchange communication pattern**. This approach has significantly decreased the communication cost, especially in sharing boundary values between **neighboring processes.**

With respect to topologies of data distribution, grid topology was preferred over linear chain and ring, because grid's communication pattern can enhance spatial locality. We have experimented both 2-by-2 and 1-by-4 grid for 4 processes to optimize the performance. One difficulty that comes with the design of 2-by-2 grid is that the correctness of data communication is hard to be achieved as we try to scale up. Therefore, we prefer the design of 1-by-4. Furthermore, it is also not surprising that halo exchange significantly help improve the correctness of the program.

Considering the solution norm, we computed the local sum across all ranks and utilized MPI_Allreduce to sum each local value to the global sum. This is important because MPI was integrated to encourage **overlap of communication and computation** so as to take advantage of intelligent communication agents and to hide communication latencies. Nonblocking communication calls were employed to separate communication initiation from completion.

Finally, to minimize execution time effectively, prioritizing communication before computation is emphasized. This ensures efficient coordination and reduces delays, contributing to overall performance improvement.

Weak Scaling



Gustafson's law

*speedup= s+p*N*

*Efficiency is calculated by T(1)/(T(N)*p)*