Question 4:

4. (Extra credit for everyone) Part of your weekly reading included a paper titled “MPI on Millions of Cores.” Given that this paper was published in 2010 (15 years ago), can you comment on what changes have occurred since 2010 that could positively and/or negatively impact our ability to fully exploit parallelism on millions of cores? Many of the papers today discuss exascale computing. Select a recent paper on exascale- computing and compare/contrast the barriers identified in the two papers that impact our ability to achieve these milestones.

The MPI on Millions of Cores discusses computing at a petascale, introducing several MPI specifications that influence performance on a large scale. They also discuss implementations of MPI that may cause better or worse performance.

In terms of scalability, the paper describes MPI’s irregular version allowing users to transfer unequal amounts of data among processes. These collectives are often used, though for large numbers of processes it takes a long time to read an entire array despite the neighbors communicating with a process staying the same. This takes place in MPI-2.2, and at the time only proposals were available for MPI-3. Another scalability issue is graph topology, where a fully distributed communication topology is advantageous at scale, though not used, as the version at the time required an entire communication graph for each process. Fault tolerance was also discussed, as MPI enables users to determine the event of an application during a fault, though it is largely non-deterministic. These interfaces also lack scalability, as they require large amounts of operations. A better approach mentioned was to allocate and return an array equal to the number of failures. Overall, the two aspects prioritized were memory consumption and performance of all collective functions including all processes. At large scale process amounts, these challenges introduce possibilities for greater inefficiency.

Since this paper, more versions of MPI have been introduced. MPI has included nonblocking collectives and more topology-aware communication. These enable greater performance amongst dynamic processes, and for greater amounts of processes overall. Accelerators for computing, including GPUs and TPUs, have allowed for better distribution of parallel workloads, and memory transfer overheads have decreased with hardware associated memory models. In addition, as previously mentioned, communication bottlenecks between processes are often a large issue when distributing data amongst 1 million cores. High-speed interconnects and more advanced network topologies, like Dragonfly, are commonly used in today’s top 10 supercomputers to enable better performance. These technologies were not nearly as efficient during the writing of the paper.

For my recent exascale computing paper, I chose “Exascale Computing: The Next Frontier of High-Performance Computing” from 2024. It discusses the state of exascale computing, including current challenges and opportunities for improving performance. The primary challenges it identifies include energy consumption, data storage and management, and lack of software optimization. It also mentions fault-tolerant systems, though in a hardware capacity, so it is somewhat similar to the previous paper. This paper doesn’t dive deep into MPI, but it does highlight more general areas. One of the primary focuses, power consumption, includes the costs of up to 100 megawatts as a projection for exascale computing, which vastly increases the current 10 to 20 MW range. MPI can be used to better optimize communication in this case, but this is not explicitly mentioned. In addition, data storage challenges are introduced at computing on exascale, as they often require petabytes or exabytes of storage. Since current memory architectures may not scale effectively, alternative approaches like high-bandwidth memory need to be implemented at scale. These problems aren’t touched upon in a greater framework in the older paper, as focus on MPI, though important, doesn’t account for all potential optimizations in creating exascale computing.

The most similar feature between the papers is their focus on fault tolerance, as this paper implores researchers to investigate fault-tolerant architectures to keep systems running during failure. The previous paper investigates fault tolerance as it relates to MPI, as users aren’t given clear procedures for how to develop safe MPI software.