Facilitating Electronic Structure Calculations on GPU-based Exascale Platforms

Date: April 12, 2023
Presented by: Jean-Luc Fattebert (Oak Ridge National Laboratory)
(The slides are available under “Materials from the Webinar” in the above link.)

Q. What were the OpenMP obstacles? Which compiler was used? How different were the library interfaces? What are the compatibility issues with OpenMP + libraries?

A. Writing a performant sparse matrix-sparse matrix multiplication in C with OpenMP pragmas does not seem achievable with current OpenMP functionalities.

Q. Can you comment on possible use of features in SYCL and any potential impacts on performance?

A. We have not investigated the possible use of SYCL and its performance in our use case.

Q. Did you compare the OpenMP offload performance across compilers?

A. We did not. We believe the performance issue we are seeing is not compiler specific, but related to the limitations of OpenMP.

Q. Have you tried the Nvidia compiler forOpenMP on Nvidia GPUs?

A. No.

C. The issues could also be the way the loops were implemented in C++ … That could contribute to performance, that could be the reason that libraries were much better …

Q. Could you comment on the scalability of your distributed matrix multiply on GPUs?

A. The code is not optimized for GPUs yet. In particular, our current MPI implementation relies on CPU to CPU communications which requires a lot of copies between GPUs and CPUs and is not optimal.

Q. What is the main obstacle to using GPU efficiently?

A. We have been trying to avoid writing non-portable code using vendor specific languages. So it left us with a few combined options: (i) use algorithms that are “GPU friendly” if possible (ii) use OpenMP for non performance-critical code (iii) use vendor libraries for performance-critical kernels for which OpenMP does not give a performance good enough.