PPPL PRINCETON PLASMA PHYSICS LABORATORY

Progress report on M3D-C1 optimization on Frontier

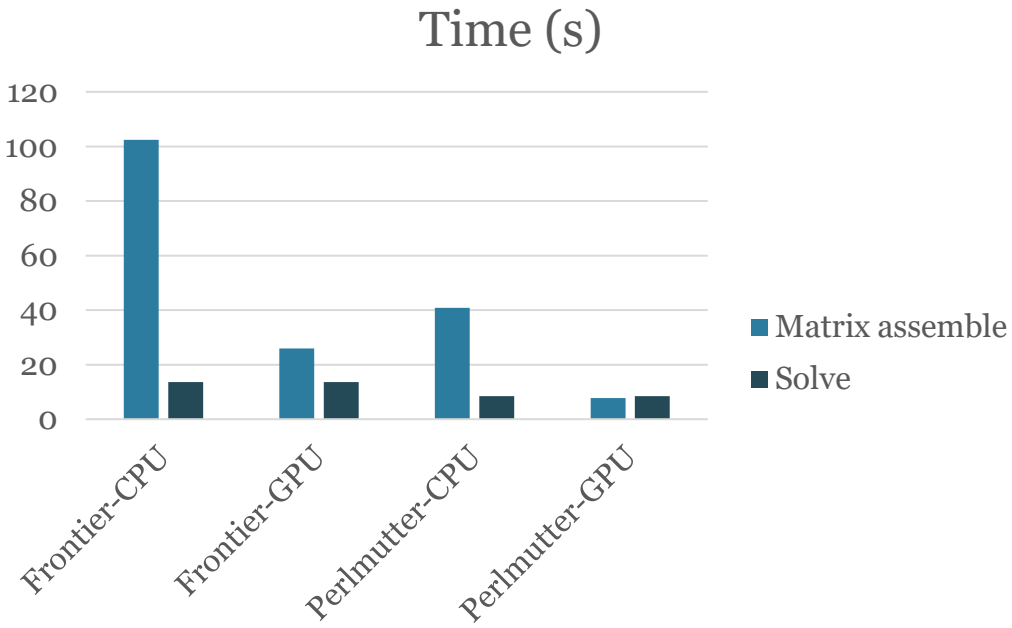
Chang Liu

Introduction to Frontier

- Frontier is the new supercomputer at OLCF, the first exascale in the world. It has 9408 AMD nodes.
  - 1 AMD 64 core CPU per node
  - 8 AMD GPUs per node
  - The CPU-GPU bandwidth is 36 GB/s, smaller compared to 50GB/s of Summit
- To compile Fortran code on Frontier with GPU support, one has to use the AMD ROCM compiler.
  - It does not support OpenACC (currently implemented in M3D-C1), only OpenMP-offloading
  - We have to do a translation of OpenACC directives to OpenMP
  - The current OpenMP directives must be disabled to avoid conflicts

Speedup of M3D-C1 matrix assembling on Frontier GPU

- Nonlinear eqsubtract=1 MHD simulation (4 planes, 5626 elements per plane)



- GPU optimization on Frontier gives about 3x speedup on matrix assembling
- Both the CPU and GPU speed seems slower on Frontier compared to Perlmutter (surprise)

Technical issues related Frontier

- The shipped system AMD ROCM compiler is an old version, and does not have good support of multiple MPI processes communicating with GPUs (though the user manual claims it has)
  - The issue is mostly resolved by upgrading to the newest version.
  - Need to compile all the dependencies by ourselves, including MPI
- Many of the OpenMP feature are not supported
  - The rewritten OpenMP code seems less efficient compared to OpenACC code
- The Fortran compiler does not support reaching global variables in GPU subroutines!
  - This means that a large part of the particle pushing code needs to be written to circumvent this issue
- We have reached Yang Chen (developer of GEM) and he said they experienced the same issue and has almost given up using OpenMP on Frontier due to the compiler bugs
  - We can wait until for Fortran support improves
  - We can rewrite the GPU part of the code in C/C++, and use Kokkos instead of OpenMP

Technical issues related to SCOREC library

- When using AMD compiler to compiler the m3dc1_scorec library, the code gives a memory error when initializing a solve matrix.
  - We tried to use GCC with -O2 and got the same issue.
  - We tried GCC with -O0 -g and the code runs well
- It may be related to the many warnings reported when compiling the m3dc1_scorec library, and the compiler may make a mistake when trying to optimize the code.

```
/ccs/home/cliu1/scorec/amd15/include/pumi_list.h:37:34: warning: 'template<class _Category, class _Tp, class _Distance, class _Pointer, class _Reference> struct std::iterator' is deprecated [-Wdeprecated-declarations]

 37 | class ListIterator : public std::iterator<std::forward_iterator_tag,T*>

/ccs/home/cliu1/scorec/m3dc1_scorec/src/m3dc1_matrix.cc: In member function 'int matrix_solve::solve(FieldID)':

/ccs/home/cliu1/scorec/m3dc1_scorec/src/m3dc1_matrix.cc:1084:1: warning: control reaches end of non-void function [-Wreturn-type]
```

- We can try to improve the C++ code quality here by using a newer version of GCC (>11) and eliminate all the warnings

Plan for next step

- We can try build a GNU compiler on Frontier with OpenACC support for AMD GPUs.
- We can contact more people (Zhihong Lin, Emily Belli) to find if there are other solutions