Activity report regarding the GPU and CPU time allocation used on Dardel in the fall of 2024

Dhrubaditya Mitra, Matthias Rheinhardt, & Axel Brandenburg (Nordita)

January 1, 2025

Summary

We have tested the performance of the Pencil Code (Pencil Code Collaboration, 2021) with GPU acceleration. It is based on the library and framework of Astaroth[1]. Both the Pencil Code and Astaroth are open source. The timing results indicate a speed-up of about 11 on *Dardel*. This is comparable to the speed-up of about 16 on *Lumi*. The present allocation to the GPU partition of *Dardel* led to a successful application for a *Large Allocation* during the last round for both *Dardel* and *Lumi*.

Academic achievements

The scaling results reported here are based on three-dimensional simulations of decaying hydromagnetic turbulence, just as in the recent papers by Brandenburg et al. (2024) and Brandenburg & Banerjee (2024). Here, we compare the Pencil Code (PC) on CPUs against the PC with Astaroth embedded (PC-A), where Astaroth integrates the partial differential equations of magnetohydrodynamics while the PC performs all peripheral tasks (diagnostics & I/O). We build with `gfortran/gcc` and `nvcc` using the modules:

```
gcc/12.2.0   craype-accel-amd-gfx90a  PDC/23.12
PrgEnv-gnu/8.5.0  cray-mpich/8.1.27   rocm/5.7.0
```

For the timing results, we routinely output with the PC both the wallclock and normalized times per time step and mesh point. We also compare with the performance on *Lumi*. On *Dardel*, we performed the following set of tests: PC-A, 8GPUs = 1 node, grid size 512^3, three repetitions:

```
 Wall clock time/timestep/meshpoint [microsec] = 1.9076658E-03
 Wall clock time/timestep/meshpoint [microsec] = 2.3275883E-03
 Wall clock time/timestep/meshpoint [microsec] = 1.9224858E-03
```

Thus, the normalized times per time step and mesh point are between 1.9 and 2.3 nanoseconds. For completeness, we also indicate here the actual wallclock times for the same three runs:

```
 Wall clock time [hours] =  7.119E-02 (+/-  5.5556E-12)
 Wall clock time [hours] =  8.687E-02 (+/-  5.5556E-12)
 Wall clock time [hours] =  7.175E-02 (+/-  8.3333E-12)
```

PC, 64 CPUs = 1 node, grid size 512^3, two repetitions:

```
 Wall clock time/timestep/meshpoint [microsec] = 2.2191811E-02
 Wall clock time/timestep/meshpoint [microsec] = 2.2515382E-02
```

Thus, without GPU acceleration, the normalized times per time step and mesh point are between 22 and 23 nanoseconds, so the comparison reveals a *speedup of about 11*. For testing weak scaling, we also performed a problem that is 8 times larger using 8 nodes instead of 1:
PC-A, 64GPUs = 8 nodes, grid size 1024^3

[1] `https://bitbucket.org/jpekkila/astaroth/`

```
 Wall clock time [hours] =  7.786E-02 (+/-  5.5556E-12)
 Wall clock time/timestep/meshpoint [microsec] = 2.6079823E-04
```

Thus, the normalized time per time step and mesh point is 0.26 nanoseconds. Here we only did one such test. Therefore, the comparison reveals roughly the same wall-clock time as with PC-A on one node and the same grid size per process. This is therefore very satisfactory.
On *Lumi*, we performed an analogous comparison:
PC-A, 1 node, 8 GPUs, grid size 380^3,

```
 Wall clock time [hours] =  1.522E-02 (+/-  8.3333E-12)
 Wall clock time/timestep/meshpoint [microsec] = 1.1079179E-03
```

Thus, the normalized time per time step and mesh point is 1.1 nanoseconds.
PC, 1 node, 64 CPUs, grid size 352^3

```
 Wall clock time [hours] =  0.192     (+/-  5.5556E-12)
 Wall clock time/timestep/meshpoint [microsec] = 1.7558866E-02
```

Thus, the normalized time per time step and mesh point is 18 nanoseconds, so the *speedup on Lumi is about 16*.

Next, we report weak scaling results:
PC-A, 1 node, 8 GPUs, grid size 512^3,

```
 Wall clock time [hours] =  4.535E-02 (+/-  5.5556E-12)
 Wall clock time/timestep/meshpoint [microsec] = 1.2150889E-03
```

PC-A, 8 nodes, 64 GPUs, grid size 1024^3,

```
 Wall clock time [hours] =  1.261E-02 (+/-  5.5556E-12)
 Wall clock time/timestep/meshpoint [microsec] = 4.2233689E-05
```

PC-A, 2048^3, 64 nodes, 512 GPUs

```
 Wall clock time [hours] =  1.510E-02 (+/-  5.5556E-12)
 Wall clock time/timestep/meshpoint [microsec] = 6.3206880E-06
```

The last two runs show roughly equal wallclock time. The longer time for the smallest grid size remains unclear.

In future, we aim to run with up to 8192^3 mesh points. To address properly the critical question of the dependence on the magnetic Reynolds number we have to use high resolution runs. As we move from 256^3 via 512^3 to 2048^3 and 4096^3 to to 8192^3 mesh points (and correspondingly higher magnetic Reynolds numbers), we expect to see the development of better scaling. To confirm our ideas and to understand the effects of what can be interpreted as magnetic reconnection, we will perform several high-resolution runs.

The present tests have also highlighted a numerical discrepancy between the GPU and CPU runs[2] that has prevented us from presenting new scientific findings based on the small allocation.

References

Brandenburg, A., & Banerjee, A., "Turbulent magnetic decay controlled by two conserved quantities," *J. Plasma Phys.*, in press, doi:10.1017/S0022377824001508, arXiv:2406.11798 (2024).

Brandenburg, A., Neronov, A., & Vazza, F., "Resistively controlled primordial magnetic turbulence decay," *Astron. Astrophys.* **687**, A186 (2024).

[2] `http://norlx65.nordita.org/~brandenb/projects/GPU-MHDdecay/`

Pencil Code Collaboration: Brandenburg, A., Johansen, A., Bourdin, P. A., Dobler, W., Lyra, W., Rheinhardt, M., Bingert, S., Haugen, N. E. L., Mee, A., Gent, F., Babkovskaia, N., Yang, C.-C., Heinemann, T., Dintrans, B., Mitra, D., Candelaresi, S., Warnecke, J., Käpylä, P. J., Schreiber, A., Chatterjee, P., Käpylä, M. J., Li, X.-Y., Krüger, J., Aarnes, J. R., Sarson, G. R., Oishi, J. S., Schober, J., Plasson, R., Sandin, C., Karchniwy, E., Rodrigues, L. F. S., Hubbard, A., Guerrero, G., Snodin, A., Losada, I. R., Pekkilä, J., & Qian, C., "The Pencil Code, a modular MPI code for partial differential equations and particles: multipurpose and multiuser-maintained," *J. Open Source Softw.* **6**, 2807 (2021).