Review report

Manuscript: *# 2019-IJFV*
Author(s): *Bérenger Bramas, Philippe Helluy, Laura Mendoza, Bruno Weber*
Title: *Optimization of a discontinuous finite element solver with openCL and starPU*

General considerations. The manuscript concerns High Performance Scientific Computing dealing with very efficient usage of the computer resource in the context of the Discontinuous Galerkin (DG) method. After a synthetic description of the DG method to settle the main concepts, the authors provide a short description of the StarPU platform and present some evidences about the efficiency of the new technology.
I give hereafter my comments and recommendations.

Comments.

1. The introduction deserves some improvements, for example
   - "the development of engineering simulation software *is difficult*" → "*faces two major antagonist issues :* one the one hand..."
   - "with a generic and *elegant*" approach → *universal*
   - "For a recent *exposition*" → *review*
   ... and many others.

2. Section 2 is quite pedagogic and provides a good overview about the DG method.

3. p.8, Algorithm. I suggest to use a left justification with indentation in place of the centered format. Indeed, it is important to easily identify the functional blocks: `if, then, else,...`.

4. Data storage and memory management is an important issue for efficient computation. Contiguous location of the data in "for loop" greatly takes advantage of the pre-fetch and cache technique whereas unstructured meshes unfortunately reduce the data compactness.
   The authors present no information about the cache missed that is a good indicator about the memory transfer bottlenecks, especially the L3-cache read/write faults. Locality of the data is also an important parameter to guarantee efficient use of the computational resource (some tilling techniques for instance). Indeed, even reading a single byte, the cache mechanism makes the CPU from loading a full line of 16 bytes hence it is of crucial importance that the other 15 bytes were used in the computational process or else we dramatically cut the bandwidth of the data flow. Do you have a compute bound or a memory bound process?

5. In GPU computation, memory transfer between host and client takes time and may represent a noticeable amount of the the total working time. Many computational processes attempt to avoid such transfer (or at least just transfer some part of the data). I hardly see how the StarPU manages the data transfer.

6. Many-core cases. Numerical simulations reported in table 2 indicate a perfect speed-up which seems to me impossible due to the overhead and memory bottlenecks. How do you justify such nice speed-up. The authors do just mention the number of DOF while the point is the amount of memory: do the data entirely fit in the L3-cache or not?
   Table 3 seems to me more realistic. Usually we represent the speed-up with respect to the single-thread process (2:1.8, 4:3.6, 8:6.9, 16:13.5) or with a curve. I have understood that you represent the speed-up from a n-core to a 2n-core configuration.

7. Do you take advantage of the vectorization (AVX2 SIMD instructions provide 4 `double` or 8 `single` arithmetic operations at the same time.)

8. The OpenMP pragma enables several kind of scheduling (variable chunk size: statict, dynamic, guided, runtime) and has a deep impact in the computational efficiency (basically we reduce the wait barrier constraint). What kind of scheduling you are using in the many-core benchmark?

9. GPU cases. Have you experiment different number of GPU-cores in the GPU benchmark to provide the speed-up of the method. You only provide the final result with (I suppose) all cores running but we have no idea about the scalability of the code. Comparing running time between two definitively different hardware is quite limited since we deal with very different configurations even with mono-thread process (memory and cache velocity, clock, NUMA in the case of the workstation).

Conclusion. The manuscript presents some interesting features for the numerical community since HPC is an important challenge and StarPU is a potential tool that could be used by the developers. I propose a minor revision following my remarks. Since the paper is mostly dedicated to the computational efficiency, the issues dedicated to the CPU/GPU optimization should be better developed and quantified.