Geophysical Research Abstracts
Vol. 19, EGU2017-7790, 2017
EGU General Assembly 2017

Accelerating 3D Elastic Wave Equations on Knights Landing based Intel Xeon Phi processors

Mohammed Sourouri and Espen Birger Raknes

Norwegian University of Science and Technology (NTNU), Trondheim, Norway (mohammed.sourouri@iet.ntnu.no, espen.raknes@ntnu.no)

In advanced imaging methods like reverse-time migration (RTM) and full waveform inversion (FWI) the elastic wave equation (EWE) is numerically solved many times to create the seismic image or the elastic parameter model update. Thus, it is essential to optimize the solution time for solving the EWE as this will have a major impact on the total computational cost in running RTM or FWI. From a computational point of view applications implementing EWEs are associated with two major challenges. The first challenge is the amount of memory-bound computations involved, while the second challenge is the execution of such computations over very large datasets.

So far, multi-core processors have not been able to tackle these two challenges, which eventually led to the adoption of accelerators such as Graphics Processing Units (GPUs). Compared to conventional CPUs, GPUs are densely populated with many floating-point units and fast memory, a type of architecture that has proven to map well to many scientific computations. Despite its architectural advantages, full-scale adoption of accelerators has yet to materialize. First, accelerators require a significant programming effort imposed by programming models such as CUDA or OpenCL. Second, accelerators come with a limited amount of memory, which also require explicit data transfers between the CPU and the accelerator over the slow PCI bus.

The second generation of the Xeon Phi processor based on the Knights Landing (KNL) architecture, promises the computational capabilities of an accelerator but require the same programming effort as traditional multi-core processors. The high computational performance is realized through many integrated cores (number of cores and tiles and memory varies with the model) organized in tiles that are connected via a 2D mesh based interconnect. In contrary to accelerators, KNL is a self-hosted system, meaning explicit data transfers over the PCI bus are no longer required. However, like most accelerators, KNL sports a memory subsystem consisting of low-level caches and 16GB of high-bandwidth MCDRAM memory. For capacity computing, up to 400GB of conventional DDR4 memory is provided. Such a strict hierarchical memory layout means that data locality is imperative if the true potential of this product is to be harnessed.

In this work, we study a series of optimizations specifically targeting KNL for our EWE based application to reduce the time-to-solution time for the following 3D model sizes in grid points: 128^3, 256^3 and 512^3. We compare the results with an optimized version for multi-core CPUs running on a dual-socket Xeon E5 2680v3 system using OpenMP. Our initial naive implementation on the KNL is roughly 20% faster than the multi-core version, but by using only one thread per core and careful memory placement using the `memkind` library, we could achieve higher speedups. Additionally, by using the MCDRAM as cache for problem sizes that are smaller than 16 GB further performance improvements were unlocked. Depending on the problem size, our overall results indicate that the KNL based system is approximately 2.2x faster than the 24-core Xeon E5 2680v3 system, with only modest changes to the code.