A proof of concept multi GPU implementation of the DFT-RTM algorithm using OpenACC directives

Átila S. Q. Soares, Senai Cimatec; Peterson Nogueira, Senai Cimatec

Abstract

The state-of-the-art family of acoustic reverse time migration (RTM) implementations for GPU generally involves wavefield reconstruction methods, such as the effective boundary and checkpoint. The first saves the boundary and the last two forward wavefields. The second save intermediate forward wavefields spaced for a time interval. Both techniques recalculate its forward wavefield at the same time that the reverse propagation is performed. Consequently, these techniques perform at least three full seismic wavefield modeling, not counting the ones necessary for the wavefields reconstruction. We propose a multi GPU implementation of an already established technique, which calculates the forward and reverse wavefields simultaneously in the time domain while concomitantly converting these wavefields to the frequency domains. This conversion is performed using a Discrete Fourier Transform kernel (DFT) along the time axis, allowing us to apply the imaging condition immediately. As a result, the wavefields from different points in time are correlated with their frequency decomposition. This is especially interesting for memory-constrained environments, where the ability to split the forward and reverse wavefield computations to a pair of GPUs is enough to solve the memory bottleneck. The latter, for instance, will hardly have enough VRAM to handle large 3D models, while it might be able to hold wavefields for the wave propagation at least. Another use case for such a technique is for imaging in attenuating media (e.g viscoacoustic and viscoelastic), given that the effective boundary technique can not be applied in those cases. For such cases, some checkpointing algorithm is generally employed, which for large 3D models can represent a massive increase in memory consumption. The physical coherence of the DFT-RTM is related to how well the DFT can correlate the time steps to a certain number of frequencies. The algorithm was implemented using the OpenACC+OpenMP directives for Fortran. A simple but fast language like Fortran and OpenACC directives allowed us to investigate how this technique can better perform in a multi-GPU environment. The Nvidia HPC SDK suite, which provides the nvfortran compiler, was used.