Maximizing Throughput for High Performance TTI-RTM: From CPU-RTM to GPU-RTM

Xinyi Sun and Sang Suh, TGS*

Summary

The implementation and production application of fast Reverse Time Migration (RTM) algorithms, especially for Tilted Transverse Isotropic (TTI) medium, is critical if the seismic industry is to quickly and efficiently create superior images for the interpretation of steep-dips and sub-salt regions. This paper presents an overview of the application of our high performance GPU-RTM algorithms developed for TTI, VTI and isotropic media. This application allows us to speed up RTM migration for large projects in Narrow Azimuth (NAZ), Wide Azimuth (WAZ) and Multi-azimuth (MAZ) (e.g. orthogonal WAZ) surveys.

Introduction

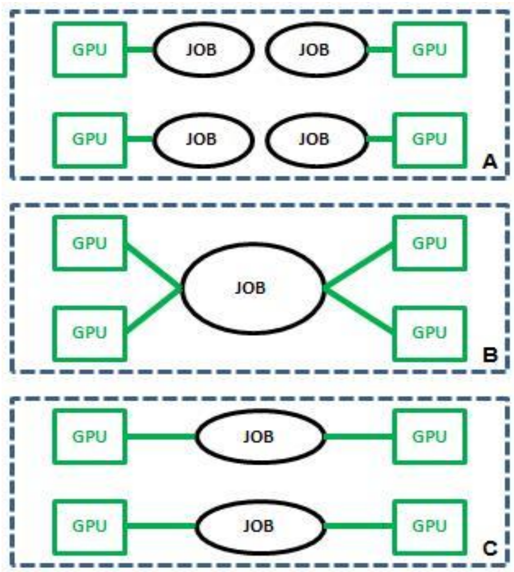
Two-way wave equation Reverse Time Migration (RTM) is computationally expensive compared with one-way Wave Equation Migration (WEM), fat-ray based Beam Migration (BM) and ray based Kirchhoff Migration (KM). Increased demand for RTM, especially TTI-RTM (Suh et al., 2010 and Yoon et al., 2010), due to its superior image quality in complex geology, has been pushing rapid development in algorithms, software and utilization of hardware.

CPU-based RTM has been well used in production on CPU-clusters with either distributive or shared memory architecture. The development and application of GPU-based RTM, for large-scale production migration has been limited (Clapp and Fu, 2010 and Foltinek et al., 2009), primarily due to limitations in software portability and hardware complexity. Continuing improvements in software and algorithm developments and recent advances in GPU-architecture have increased the throughput and the price-performance ratio of GPU based clusters, making GPU-RTM the preferred choice in light of current hardware trends.

We have explored the features of the latest GPU architecture on the market from NVIDIA and ported our CPU-RTM algorithms for TTI, VTI and isotropic (ISO) media to GPU clusters and have moved from the original multi-core parallelization of CPU-RTM to the super-parallelism of GPU-RTM. To fully utilize both GPU and CPU resources, especially for TTI-RTM, we have also taken into consideration the diversity of the applications (e.g. NAZ or WAZ migration, model-building or final migration) in order to optimize the load balancing through a combination of different job-submission configurations.

GPU-RTM Methodology

The extension of CPU-RTM to GPU-RTM follows the trend of computer technology development from main-frame to cluster and from single-CPU to multi-CPU. The strategy of divide-and-conquer applies to seismic applications where parallel computations dominate the entire algorithm of an application. While the number of CPUs per node is limited (e.g. 4-16 cores per-node in normal production clusters), the number of cores per GPU-unit can be large (e.g. 512 CUDA cores in NVIDIA's Fermi architecture). Setting up each node of a GPU cluster with multi-CPU and multi-GPU options provides GPU-RTM coding the flexibility to fully utilize GPU and CPU resources via a range of job configuration options tailored to different algorithms (TTI, VTI or ISO) and different types of projects (e.g. NAZ or WAZ, model-building or final migration stage).



A: 4x1 job　**B: 1x4 job**　**C: 2x2 job**

Figure 1: GPU-RTM job configurations.

Job configuration possibilities include:

1. Multi-job and single-GPU (Figure 1A): For this setup the GPU-RTM could run multiple jobs on each node. Each job uses a single GPU-unit with

dedicated GPU resources and shared CPU resources.

2. Single-job and multi-GPU (Figure 1B): using all GPU and CPU resources with the domain-decomposition technique, a single job could be submitted on each node to run a large TTI-RTM project.

3. Multi-job and multi-GPU (Figure 1C): where both GPU and CPU resources are shared to fulfill a medium-size computational task.

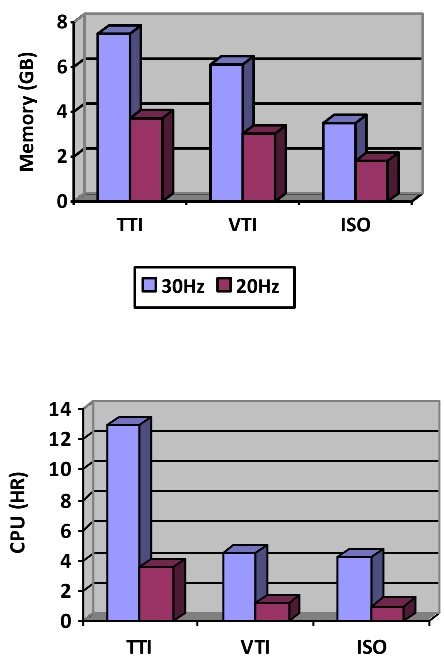


Figure 2: CPU-RTM memory and runtime.

GPU-RTM Application

The computational cost for RTM is directly related to the Number-Crunching-Size (NCS) possessed by a digitization-volume on which wave-propagations from both source and receiver arrays are simulated. For Finite Difference quasi-acoustic implementation of GPU-RTM the NCS is a function of several variables, including:

NCS (Algorithm, Aperture, Zmax, Vmin, Fmax, Tmax)

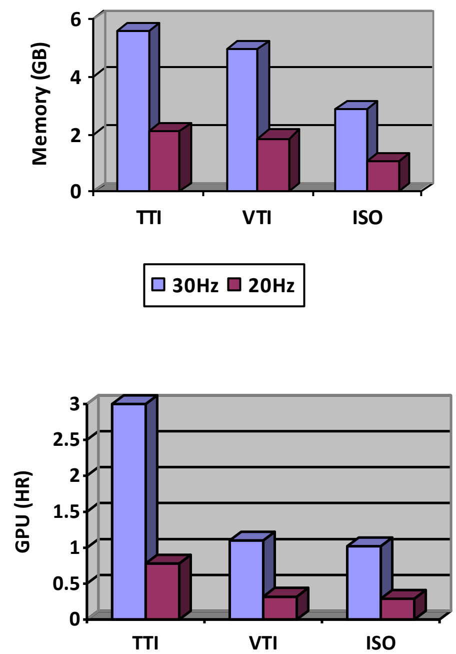


Figure 3: GPU-RTM (4x1) memory and runtime.

1. Domain-size: Our GPU-RTM is implemented in the shot-domain. RTM migration aperture (***Xmax*** and ***Ymax***), migration depth (***Zmax***) and recording time (***Tmax***) are the first factors to be considered in terms of computational time and resource usage. Depending on the geological complexity of each project, migration aperture could be small (8x8km), m./edium (9x9km), large (12x12km) or very large (16x16km). Migration depth, which is controlled by the target depth and stage of a project, could be shallow (<10km), medium (10-16km) or deep (16-20km).

2. Grid-size: Division of the domain size by the grid-size gives an NCS-volume which could be very large with a very small grid size. Both temporal and spatial grid sizes of the digitization volume (***dt, dx, dy*** and ***d***z) are controlled by the RTM model's minimum velocity (***Vmin***) and the input data's maximum frequency (***Fmax***). The frequency of a normal RTM job could be low (~20Hz), high (>30Hz) or very high (>50Hz). Furthermore a model's minimum velocity also

depends on whether the RTM job is run with re-datuming or without re-datuming e.g. using RTM layer-stripping capability (Wang et al., 2011).

3. RTM-algorithm: The major driving factor for developing the GPU-RTM is the resource-intensive nature of advanced TTI-RTM algorithm with its large computational time and memory requirements.

Figure 2 illustrates the CPU-time and memory dependency of RTM algorithms with 30Hz and 20Hz maximum frequency migration. Figure 3 shows the same set of migration parameters as in Figure 2 but for GPU's resource requirement.

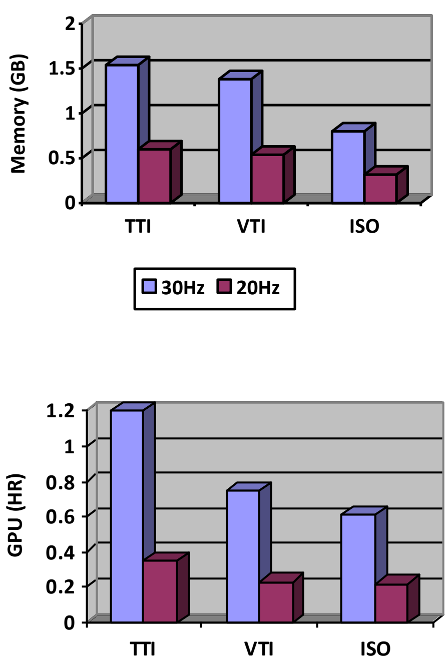


Figure 4: GPU-RTM (1x4) memory and runtime.

For a given RTM project the multi-GPU set-up of each node exploits the highly-parallel nature of RTM algorithms and uses domain-decomposition across GPU-units plus a massive number of cores within the GPU-unit, to provide the optimal loading balance through job submission configuration. Figure 4 shows a single-job and multi-GPU (4x1) configuration where memory resource is shared to fulfill the task without sacrificing performance.

GPU-RTM Examples

We have been using production GPU-RTM on our projects that require quick turn-around, and multiple iterations at the model-building stage for TTI-RTM.

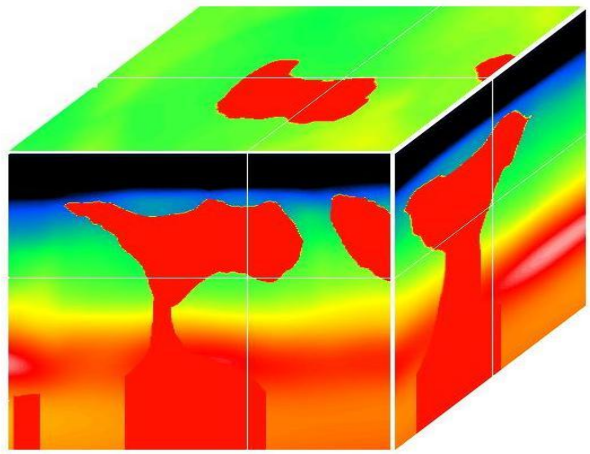


Figure 5: MAZ (orthogonal WAZ) intermediate velocity model (TOS2).

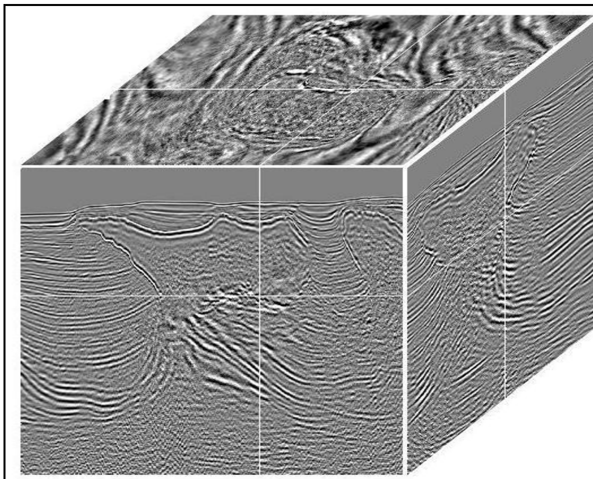


Figure 6: CPU-RTM for MAZ survey-K (TOS2).

To compare the performance of GPU-RTM over CPU-RTM, an illumination study was performed on an orthogonal WAZ project during model-building stage for the second top of salt (Figure 5). Survey-K is completed with CPU-RTM and survey-J (acquired in a shooting direction orthogonal to that of survey-K) is completed with GPU-RTM.

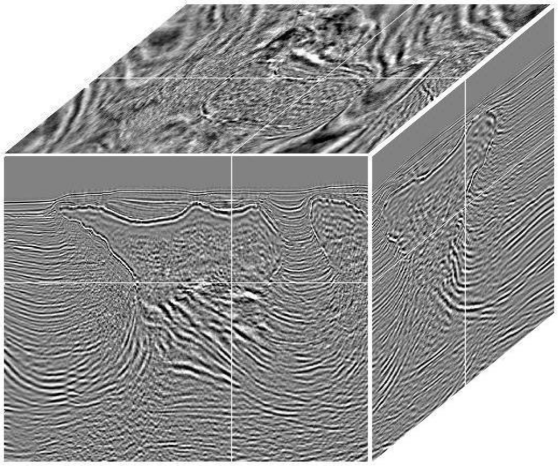


Figure 7: GPU-RTM for MAZ survey-J (TOS2).

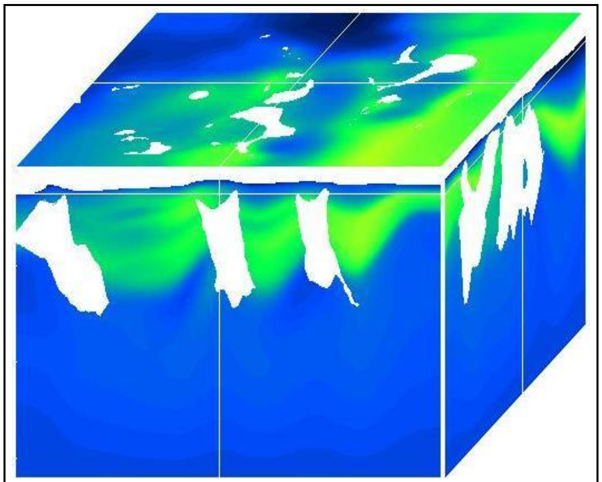


Figure 8: Anisotropic model of GPU-RTM.

The CPU-RTM ran on a 12-core per-node cluster (Figure 6), with performance as shown in Figure 2, and the GPU-RTM ran on a four-card per-node GPU-rack with a 4x1 job configuration (Figure 7).

Compared with small projects a significant factor of speed-up using GPU-RTM over CPU-RTM for large TTI-RTM projects, especially at the final migration stage (e.g. Figure 8), could dramatically shorten the project delivery time.

Conclusions

We have expanded our CPU-RTM to GPU-RTM to meet an increasing demand for RTM, especially TTI-RTM, on large-scale production projects. A favorable price-performance ratio makes GPU-RTM the preferred choice to speed-up project turn-around and to reduce production cost.

To maximize throughput for each project, especially for TTI-RTM projects, our GPU-RTM code has fully utilized both GPU and CPU resources by taking advantage of a multi-GPU and multi-CPU in the node setup and our GPU-RTM job-submission configuration has been optimized in load balancing based on parameters of each project.

An RTM with faster turn-around provides the seismic industry with more tools to attack tough imaging problems like in sub-salt areas, and for example, the RTM angle-gather (Yoon et al., 2011) and Delayed Image Time (DIT) gathers (Whiteside et al., 2011). Field trials and production experience have consistently demonstrated the superior performance of GPU-RTM over CPU-RTM.

Acknowledgments

The authors would like to thank Zhiming Li, Bin Wang, Gary Rodriguez, Terry Hart and Xuening Ma of TGS Imaging Services for leading the GPU-RTM production project, James Cai and Alex Yeh for TGS R&D, Nick Cato and Bruce Meyers for TGS IT support. We also thank Laurie Geiger, Simon Baldock and Chuck Mason for reviewing and proof-reading this paper. Finally, we wish to thank TGS for permission to publish this paper.

EDITED REFERENCES

Note: This reference list is a copy-edited version of the reference list submitted by the author. Reference lists for the 2011 SEG Technical Program Expanded Abstracts have been copy edited so that references provided with the online metadata for each paper will achieve a high degree of linking to cited sources that appear on the Web.

REFERENCES

Clapp, R. G., and H. Fu, 2010, Selecting the right hardware for reverse-time migration: The Leading Edge, **29**, 48–58.

Foltinek, D., D. Eaton, J. Mahovsky, P. Moghaddam, and R. McGarry, 2009, Industrial-scale reverse-time migration on GPU hardware: 79th Annual International Meeting, SEG, Expanded Abstracts, 2789–2793.

Suh, S., A. Yeh, B. Wang, J. Cai, K. Yoon, and Z. Li, 2010, Cluster programming for reverse-time migration: The Leading Edge, **29**, 94–97.

Wang, B., J. Ji, K. Yoon, J. Cai, W. Whiteside, M. Mason, and Z. Li, 2011, Layer-stripping RTM based on wavefield redatuming: Presented at the 81st Annual International Meeting, SEG.

Whiteside, W., Z. Guo, and B. Wang, 2011, Automatic RTM-based DIT scan picking for enhanced salt interpretation: Presented at the 81st Annual International Meeting, SEG.

Yoon, K., M. Guo, J. Cai, and B. Wang, 2011, 3D RTM angle gathers from source wave propagation angle and dip of reflector: Presented at the 81st Annual International Meeting, SEG.

Yoon, K., S. Suh, J. Ji, J. Cai, and B. Wang, 2010, Stability and speedup issues in TTI RTM implementation: 80th Annual International Meeting, SEG, Expanded Abstracts, 3221–3224.