HP14

Performance Characterization of a Vector Architecture for Seismic Applications

V. Etienne[1], A. Momin[1], L. Gatineau[2], S. Momose[2]

[1] Saudi Aramco, EXPEC ARC; [2] NEC Deutschland GmbH

Summary

Explicit time-domain finite-difference (TD-FD) methods are largely used in seismic exploration. They are at the heart of wave-equation based geophysical algorithms such as Reverse Time Migration and Full Waveform Inversion. Due to the ever-increasing amount of acquired seismic data and the need for higher resolution to optimize oil production, it is crucial to deploy TD-FD on High Performance Computing (HPC) platforms. In this work, we explore the performance reachable on vector architectures. The study is done on a traditional scalar CPU to get a performance baseline, and on a vector solution which was heavily used in the past by the O&G industry.

Introduction

Explicit time-domain finite-difference (TD-FD) methods are largely used in seismic exploration. They are at the heart of wave-equation based geophysical algorithms such as Reverse Time Migration and Full Waveform Inversion. Due to the ever-increasing amount of acquired seismic data and the need for higher resolution to optimize oil production, it is crucial to deploy TD-FD on High Performance Computing (HPC) platforms. In this work, we explore the performance reachable on vector architectures. The study is done on a traditional scalar CPU to get a performance baseline, and on a vector solution which was heavily used in the past by the O&G industry (Levin, 1986; Cheshire et al., 1987; Kampe et al., 1988; Young et al., 1991; Yamada and Momose, 2018; Mathur et al., 2020).

Methodology

Although there exist generic HPC benchmarks to measure key performance indicators of hardware components such as the STREAM benchmark (McCalpin, 1995) for the memory bandwidth or the OSU Micro benchmarks (Bureddy et al., 2012) for the MPI communications, there are few codes appropriate to benchmark geophysical applications. In general, exploration geophysicists do performance measurements and optimization tuning on proprietary codes. Consequently, performances are impacted by pre-defined technical choices. In this work, we take a step back and characterize individually key components of a seismic algorithm, with a focus on the TD-FD. With the recently developed benchmark HPCscan (https://github.com/vetienne74/hpcscan), a code that implements scientific kernels in C++, we characterize the memory bandwidth, the finite-difference stencil computations and the scalability of an acoustic propagator on two modern architectures with the specifications described below.

Table 1 *Specifications of the systems benchmarked in this study*

Product	Architecture	Cores	Memory	Peak compute perf.	Interconnect
A1	Scalar	2 sockets x 20 cores	DDR4 Max. 281 GB/s	Peak single precision 5.89 TFlop/s	Ethernet 10 Gbit/s
A2	Vector	8 cores	HBM2 Max. 1351 GB/s	Peak single precision 4.30 TFlop/s	InfiniBand 100 Gbit/s

All tests are performed in single precision with 3D grids ranging from 4 to 32 GB, representative of seismic applications. Up to 8 computing nodes or 8 accelerators are used.

Benchmark of the memory bandwidth

One of the key performance indicators of a computing platform is the memory bandwidth. HPCscan embeds a test case similar to the STREAM benchmark. The results for this test case are presented in Fig. 1. The left figure shows the sustained memory bandwidth measured on product A1 for 5 simple array operations when the number of threads is increased from 1 to 40, matching the total number of physical cores available on the node. We observe that the highest performance is achieved with the 'Add and Update' function (considering 2 arrays A and B, this function is defined as A=A+B) with a maximum of 216 GB/s (77 % of peak). Less efficient is the function 'Fill' (A is initialized with a constant value) that performs at about half of the 'Add and Update' bandwidth. We can also detect NUMA (Non Uniform Memory Access) effects that translate into an exponential increase when threads are filling the 1st socket (from 1 to 20) and a second linear regime on the second socket (from 21 to 40). Right figure shows the bandwidth for product A2. On this device, all functions behave nearly the same with a maximum of 1085 GB/s (80% of peak) and a uniform scalability is observed (no NUMA effect since memory access is uniform for all the 8 cores). Overall, for simple operations on arrays, the vector architecture allows to reach a speedup between 5x and 8x vs the scalar architecture.

Benchmark of the finite-difference operators

In this section, we investigate the performance of FD stencil operations which are intensively used in seismic applications for wave modelling. It is possible with HPCscan to measure the bandwidth of FD operators such as 2nd derivatives in space (unidirectional and Laplacian FD operators). Fig. 2 presents the results obtained on A1 in Gpoint/s and GB/s, left and right respectively (GFlop/s are also measured but not displayed here).

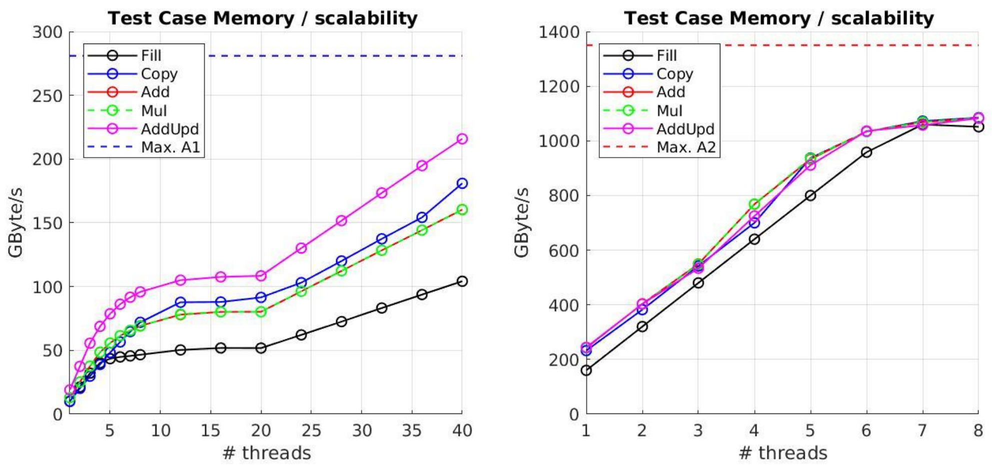

***Figure 1** Memory bandwidth benchmark. Left: product A1, Right: product A2.*

Different FD orders in space are evaluated (horizontal axis: O2, O4, O8, O12 and O16). Unidirectional derivatives along the 1st, 2nd, and 3rd axis and Laplacian operator are represented in black, blue, green and pink colours respectively. Moreover, a standard implementation of these operators (Baseline) is compared against an optimized cache blocking implementation which is a standard method to accelerate stencil computations on CPUs (Etienne et al., 2017). To facilitate the comparison between platforms, we consider the average GPoint/s measured in Fig. 2. In average, the Baseline implementation performs at 11.8 GPoint/s with a maximum of 458 GFlop/s (8% of peak compute). The cache blocking implementation performs at 14.6 GPoint/s with a maximum of 659 GFlop/s (11% of peak). The beneficial effect of the cache blocking is clearly seen on the right figure. The continuous green line that represents the derivative along 3rd axis (Baseline) is slightly below the maximum bandwidth measured in the previous memory bandwidth benchmark (horizontal blue line). The kernel using the cache blocking algorithm (dashed green line) gets a major boost with a maximum bandwidth around 800 GB/s. The maximum observed bandwidth is 1616 GB/s, which is 5.8x the peak memory bandwidth indicating a large amount of data reuse (data in-cache effects).

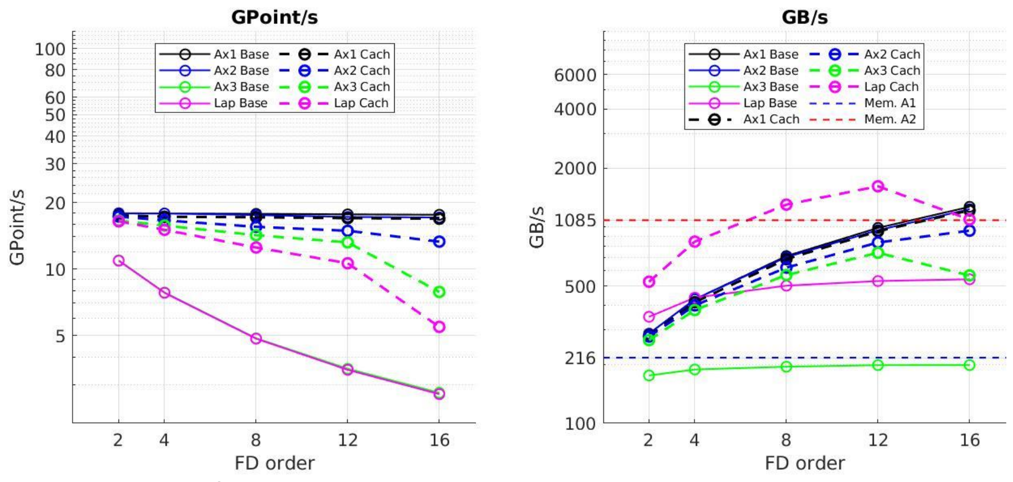

***Figure 2** Bandwidth of FD operators on product A1. Continuous lines: Baseline implementation, dashed lines: Cache Blocking implementation. Left: Gpoint/s. Right: GB/s.*

Fig. 3 shows the results obtained on the product A2 when the Baseline and Cache Blocking implementations are executed on this system without any code modification. In average, the Baseline implementation performs at 29.7 GPoint/s (2x vs A1) with a maximum of 551 GFlop/s (13% of peak). The Cache Blocking implementation performs at 30.4 GPoint/s (2.1x vs A1) with a maximum of 522 GFlop/s (12% of peak). At maximum the measured bandwidth (1316 GB/s) is about the peak memory bandwidth, indicating that data reuse is low. However, with no code modification, the vector architecture readily provides a 2x speedup compared to a kernel executed on and optimized for the architecture A1.

To increase performances, we created 2 new implementations of the kernel. The first one relies on A2 compiler pragmas and the second one on a proprietary mathematical library dedicated to stencil operations for A2. Fig. 4 presents the performance obtained with these 2 customized implementations.

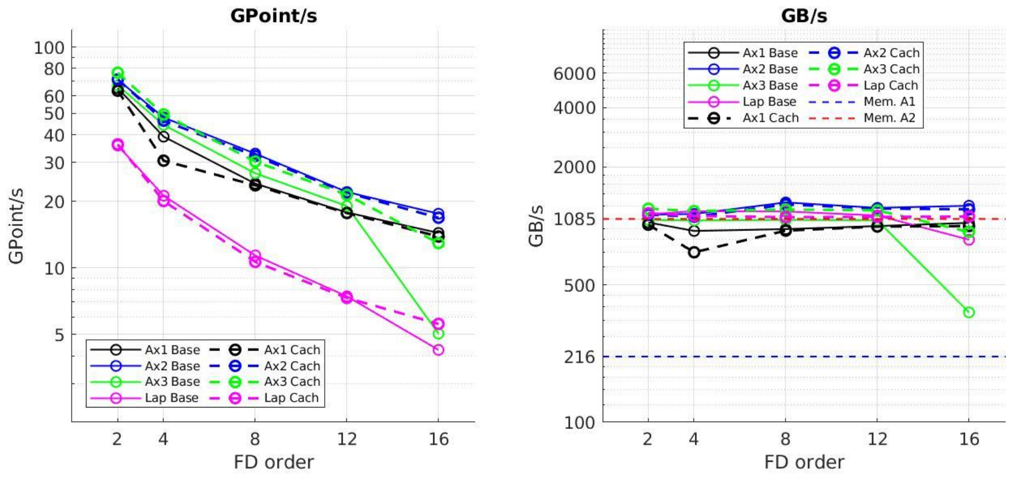

***Figure 3** Same as Figure 2 with product A2.*

The first customized implementation (Custom1) allows to reach an average of 53.8 GPoint/s (3.7x vs A1) with a maximum of 1588 GFlop/s (37% of the peak). A significant higher performance is achieved with the second implementation (Custom2) that provides in average 77.9 GPoint/s (5.3x vs A1) and a maximum of 2633 GFlop/s (61% of the peak). The highest measured FD bandwidth is 7294 GB/s which is 5.4x the peak memory bandwidth indicating a large amount of data reuse.

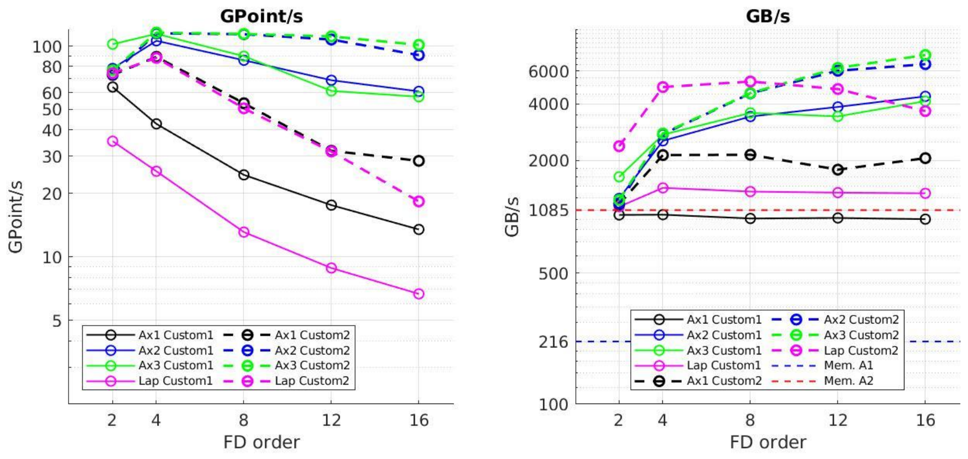

***Figure 4** Same as Figure 3 with customized implementations.*

Performance and scalability of the acoustic TD-FD propagator

Below, the performance and scalability of the acoustic wave propagator (isotropic 2nd order wave equation) are analysed. Using a subdomain decomposition method with MPI communication, strong and weak scalability analysis are conducted with a maximum of 8 subdomains (with one product A1 or A2 per domain). For the strong scalability, the problem size is fixed (1000x1000x1000) while for the weak scalability, the problem size grows linearly with the number of subdomains (from 1000x1000x1000 to 1000x1000x8000). Fig. 5 shows the results obtained on an A1-based cluster with inter-node communication via Ethernet 10 Gbit/s (blue lines) and on an A2-based system with an Infiniband 100 Gbit/s interconnect for the communication (red lines). On the A1-based system, we observe a 2.3x speedup for the strong scalability (8.7 to 20.1 Gpoint/s) and 4.3x for the weak scalability (8.7 to 37.1 Gpoint/s). As expected, higher speedup is obtained for the weak scalability due to a lower ratio of communication versus computation compared to the strong scalability case. The relative low bandwidth of the interconnect (10 Gbit/s) affects significantly the strong scalability. On A2-based system, we observe a 4.4x speedup for the strong scalability (24.7 to 108 Gpoint/s) and 5.7x for the weak scalability (24.7 to 142 Gpoint/s). A summary of performance comparisons is provided below.

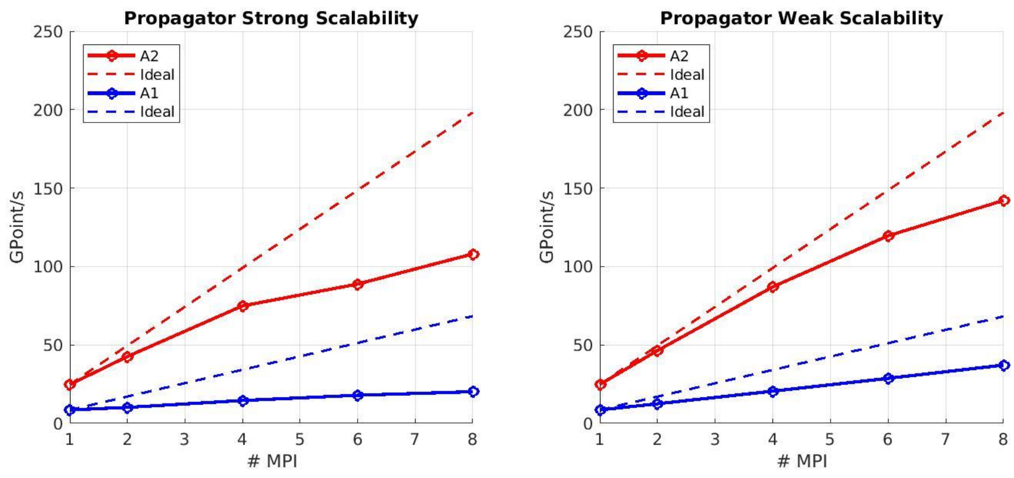

Figure 5 *Scalability analysis of the acoustic propagator. Left: strong scalability, right: weak.*

Conclusions

Performances obtained with the vector architecture A2 were compared against a traditional scalar architecture A1. A2 features a higher memory bandwidth (1.35 TB/s) thanks to the HBM2 (High Bandwidth Memory) compared to the DDR4 memory (281 GB/s) of the product A1. The vector architecture A2 is almost transparent for the C++ programmer and allows to get 5x to 8x speedup on simple operations on arrays and 2x on finite-difference stencil operations without modification of existing C++ codes. For optimal performances, efficient vectorization may require some adjustments. Especially, for stencil kernels, the dedicated mathematical library allows to reach higher performances. For the acoustic propagator application, speedups are as follows:

- One product A2 vs one product A1 ⇒ 2.8x speedup
- One node with 8 A2 vs one A1 node ⇒ 13-17x speedup
- One node with 8 A2 vs 8 A1 nodes ⇒ 3.8-5.4x speedup (but interconnect Ethernet 10 Gbit/s affects dramatically the performance of the A1-based cluster)

This work highlights the benefits of vector architectures for seismic applications. Further investigations will focus on more complex kernels than the simple acoustic propagator. Performances of FD stencil operations presented in this work suggest that higher speedups might be achieved.

References

Bureddy, D., Wang, H., Venkatesh, A., Potluri, S. and Panda, D.K., 2012, September. Omb-gpu: A micro-benchmark suite for evaluating mpi libraries on gpu clusters. In *European MPI Users' Group Meeting* (pp. 110-120). Springer, Berlin, Heidelberg.

Cheshire, I.M. and Pollard, R.K., 1987, July. Advanced Numerical Techniques for Reservoir Simulation and their Use on Vector and Parallel Processors. In *Mathematics in Oil Production* (pp. cp-235). European Association of Geoscientists & Engineers.

Etienne, V., Tonellot, T., Malas, T., Ltaief, H., Kortas, S., Thierry, P. and Keyes, D., 2017, October. High-performance seismic modeling with finite-difference using spatial and temporal cache blocking. In *Third EAGE Workshop on High Performance Computing for Upstream* (Vol. 2017, No. 1, pp. 1-5). European Association of Geoscientists & Engineers.

Kampe, F.C. and Nguyen, T.M., 1988. Performance comparison of the CRAY-2 and CRAY X-MP on a class of seismic data processing algorithms. *Parallel computing*, *7*(1), pp.41-53.

Levin, S.A., 1986. Life beyond the vector. *The Leading Edge*, *5*(12), pp.17-18.

Mathur, R., Atcheson, R. and Kubo, Y., 2020. Optimizations for seismic applications on the NEC SX-Aurora TSUBASA. In *SEG Technical Program Expanded Abstracts 2020* (pp. 2850-2852). Society of Exploration Geophysicists.

McCalpin, J.D., 1995. *STREAM benchmark. Link: www.* cs. virginia. edu/stream/ref. html# what, 22.

Yamada, Y. and Momose, S., 2018, August. Vector engine processor of NEC's brand-new supercomputer SX-Aurora TSUBASA. In *Proceedings of A Symposium on High Performance Chips, Hot Chips* (Vol. 30, pp. 19-21).

Young, L.C. and Zarantonello, S.E., 1991, August. High performance vector processing in reservoir simulation. In *Proceedings of the 1991 ACM/IEEE conference on Supercomputing* (pp. 304-315).