Introduction

The 3D acoustic wave equation is widely used in its second order formulation for seismic modeling (Alford et al., 1974), seismic imaging (Chang and McMechan, 1990) and seismic inversion (Virieux and Operto, 2009; Zhang and Luo, 2013). Despite being computationally more expensive than the widely used second order formulation, solving the first order formulation has several advantages such as allowing the use of more advanced incidence angle based imaging conditions for the RTM and the generation of angle gathers. In fact, Poynting vector computation is straightforward with this form. This paper revisits the high performance implementations of the first order formulation of the 3D acoustic wave equation by assessing the performance impacts of two cache blocking techniques: spatial blocking (SB) and multicore wavefront diamond temporal blocking (MWD-TB) techniques.

FDTD for the first order formulation of the acoustic wave equation

In its second order formulation, the 3D acoustic wave equation is written as:

$$\frac{1}{V_P{}^2(\mathbf{x})}\frac{\partial^2 P(\mathbf{x},t)}{\partial t^2} = \frac{\partial^2 P(\mathbf{x},t)}{\partial x^2}+\frac{\partial^2 P(\mathbf{x},t)}{\partial y^2}+\frac{\partial^2 P(\mathbf{x},t)}{\partial z^2}+\delta(\mathbf{x}-\mathbf{x_s})s(t), \tag{1}$$

where V_P is the velocity of the P-wave that propagates in the medium and P is the pressure, both at position $\mathbf{x}(x,y,z)$, and $s(t)$ is the source term injected at $\mathbf{x_s}$. The density is assumed homogeneous (equal to 1 kg/m3). This equation can be re-written as a system of first order PDEs :

$$\begin{aligned}
\frac{1}{V_P{}^2(\mathbf{x})}\frac{\partial P(\mathbf{x},t)}{\partial t} &= \frac{\partial v_x(\mathbf{x},t)}{\partial x}+\frac{\partial v_y(\mathbf{x},t)}{\partial y}+\frac{\partial v_z(\mathbf{x},t)}{\partial z}+\delta(\mathbf{x}-\mathbf{x_s})s(t)\\
\frac{\partial v_x(\mathbf{x},t)}{\partial t} &= \frac{\partial P(\mathbf{x},t)}{\partial x}\\
\frac{\partial v_y(\mathbf{x},t)}{\partial t} &= \frac{\partial P(\mathbf{x},t)}{\partial y}\\
\frac{\partial v_z(\mathbf{x},t)}{\partial t} &= \frac{\partial P(\mathbf{x},t)}{\partial z},
\end{aligned} \tag{2}$$

where v_x, v_y and v_z denote the particle velocities along the three spatial axes at the position $\mathbf{x}(x,y,z)$. To discretize the set of Equations (2), velocity components are computed at half discretization points of the pressure field both in time and in space. The resulting system to solve the set of Equations (2) for an 8th order in space and 2nd order in time discretization is detailed in Equation (3) where Δt denotes the spatial sampling and O_i^8 represents the 8th order spatial operator to evaluate the spatial first derivative along index i. This discretized system is used to advance the solution in time. For each time iteration, a first sweep over the domain updates the velocity wavefield, and a second sweep updates the pressure wavefield.

$$\begin{aligned}
P_{i,j,k}^{n+\frac{1}{2}} &= P_{i,j,k}^{n-\frac{1}{2}}+V_{i,j,k}^2\Delta t[O_i^8(Vx_{i,j,k}^n)+O_j^8(Vy_{i,j,k}^n)+O_k^8(Vz_{i,j,k}^n)]\\
Vx_{i+\frac{1}{2},j,k}^{n+1} &= Vx^n+\Delta t\ O_i^8(P_{i+\frac{1}{2},j,k}^{n+\frac{1}{2}})\\
Vy_{i,j+\frac{1}{2},k}^{n+1} &= Vy^n+\Delta t\ O_j^8(P_{i,j+\frac{1}{2},k}^{n+\frac{1}{2}})\\
Vz_{i,j,k+\frac{1}{2}}^{n+1} &= Vz^n+\Delta t\ O_k^8(P_{i,j,k+\frac{1}{2}}^{n+\frac{1}{2}})
\end{aligned} \tag{3}$$

Cache blocking strategies

Spatial cache blocking

We provide an initial reference implementation that enhances in-memory data reuse by relying on spatial cache blocking (SB). A brute force search is performed to assess the best cache blocking sizes for x and y dimensions. Results are reported in Figure 1 and show that the best performance is achieved for a

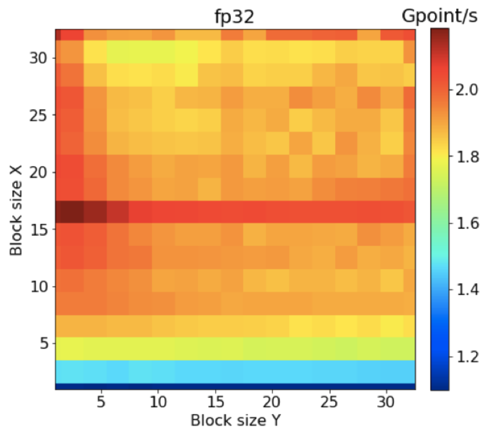

Figure 1 *SB implementation performance for different cache block sizes.*

16×4 cache block size. Figure 3 shows the roofline model for the performance of the two most time-consuming loops (pressure and velocity updates) with and without spatial cache blocking. In both cases, OpenMP threads are used to divide work among the physical cores of the considered platform (dual-socket 16-core Intel Haswell E5-2698). Table 1 details the aggregated memory traffic and arithmetic intensity (AI) as well as the total running time of the most time consuming loop (velocity components update) for each implementation using a 512^3 domain size and 200 time iterations. In particular, the table shows the bandwidth breakdown for each level of the memory system, i.e., L1, L2 and L3 cache levels and DRAM. Both figures show that L3 and DRAM traffic are very close when SB is disabled. This means that SB limits L3 cache thrashing. This results in a 4x overall speedup.

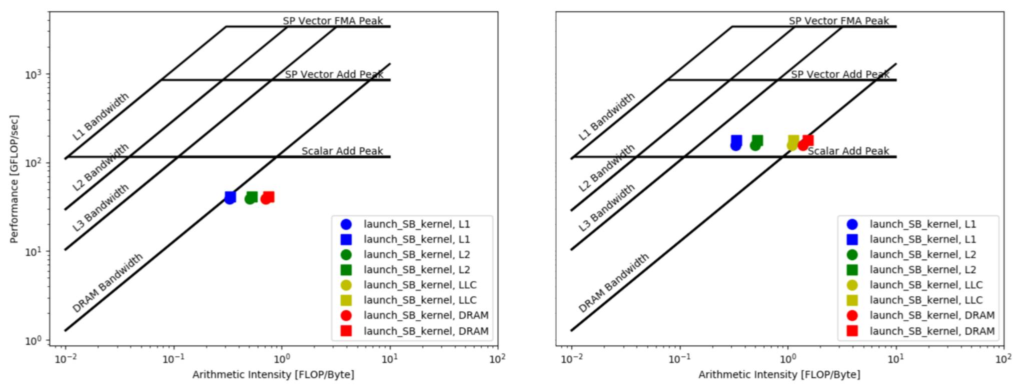

Figure 2 *Roofline model for velocity and pressure update loops for the naïve (left) and SB (right) versions.*

Kernel	**L1**		**L2**		**L3**		**DRAM**		**TIME**
	Memory GB	AI	Memory GB	AI	Memory GB	AI	Memory GB	AI	
No CB	3194.4	0.32	2071.1	0.5	1479.5	0.7	1475	0.7	24.25
SB	3210.35	0.33	2106.59	0.5	962.9	1.1	769.3	1.37	6.63
MWD-TB	3248.06	0.32	2042.44	0.5	1458.5	0.72	360.16	2.9	4.09

Table 1 *Aggregated memory traffic and arithmetic intensity (AI) for each level of the memory system (L1, L2 and L3 cache levels and DRAM) as well as the total running time of the most time consuming loop (velocity components update) for each implementation using a* 512^3 *domain size and 200 time iterations.*

Temporal cache blocking

We investigate the extension of Multicore-optimized Wavefront Diamond Temporal Blocking (MWD-TB) approach (Malas et al., 2015, 2017) - already introduced for the second-order differential formulation (Etienne et al., 2017) - to the first order formulation. This is achieved by doubling the total number of time iterations and performing an update of the pressure wavefield and the velocity wavefield alternately (Figure 3). This adaptation is possible since the stencil radius of the first order formulation (3.5 grid points) is smaller than the second order form (4 grid points). This enables to maintain the data dependencies using the same parallelization scheme.

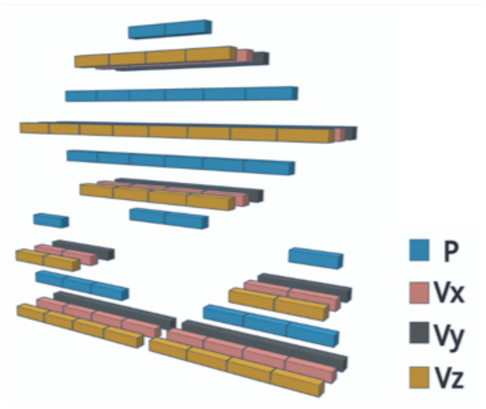

Figure 3 *MWD-TB diamond representation for the first order formulation. Pressure and velocity wavefields are updated alternately.*

Similarly to the results described above (naïve and SB cases), Figure 4 and Table 1 show the impact of introducing MWD-TB implementation on the performance of the two most time consuming loops. Results show how MWD-TB releases even further the pressure on the DRAM memory largely increasing its associated Arithmetic Intensity and reducing by more than a half the overall traffic compared to SB. This comes at the cost of more data traffic to the L3 cache. The three implementations have very similar impacts on the L1 and L2 cache levels that are not shared in the Haswell architecture.

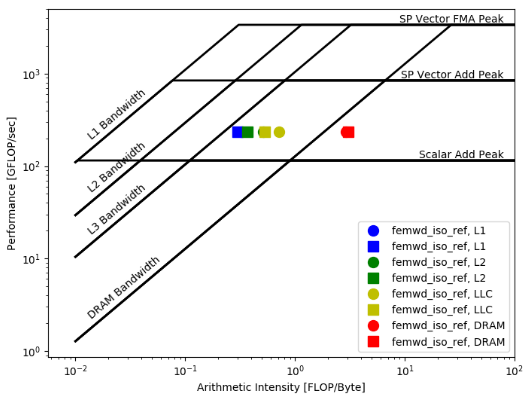

Figure 4 *Roofline model for MWD-TB velocity and pressure update loop.*

Scalability results reported in Figure 5 compare the behavior of the MWD-TB implementation and the SB implementation with two thread affinity types (compact and scatter) when the number of cores and OpenMP threads increases. We recall that the considered Haswell platform (one Shaheen node) is a 16-core dual sockets. The SB implementation with compact pinning (i.e., threads are placed as close to each other as possible) quickly saturates the bandwidth of the first socket and its performance plateaus between 8 and 16 threads at about 90 GFLOP/s. The performance increases then linearly up to 32 threads achieving up to 180 GFLOP/s. Adapting an opposite affinity strategy with the scatter affinity type (i.e., threads are placed as far as possible from each other), the plateau is reached starting from 16 threads achieving close to the maximum platform performance for SB. The MWD-TB implementation has its own pinning strategy, placing the threads of the same diamond as close as possible to each other to benefit from data locality and distributing diamonds among processors to exploit memory bandwidth. Starting from 16 threads, Figure 5 shows that MWD-TB outperforms the two former implementations. For MWD-TB, there is a trade-off to find between the number of threads per diamond and the number of diamonds. For this initial assessment, we report results for a simple strategy creating diamonds of one thread when the total number of threads is below four and diamonds of four threads otherwise. Better tuned strategies will be explored as future work and may improve MWD-TB performance for small numbers of threads.

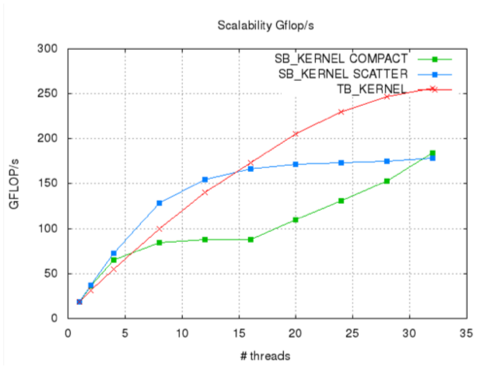

Figure 5 *Scalability on Haswell.*

Conclusions

We present two optimizations techniques for the first order formulation of the 3D acoustic wave equation, which intend to maximize memory bandwidth across the memory subsystem. Spatial blocking (SB) permits to increase data reuse from main memory. To alleviate memory pressure, the multicore wavefront diamond temporal blocking (MWD-TB) technique levels the pressure up to last level cache, cutting down significantly data movement to main memory. SB achieves 6X performance speedup against the reference implementation (no cache blocking). The MWD-TB implementation highlights its performance superiority against SB and achieves up to 1.5X performance speedup. For future work, we would like to investigate the performance impact of integrating the Convolutional Perfectly Matched Layer (CPML) formulation (Komatitsch and Martin (2007)) as boundary conditions into MWD-TB.

References

Alford, R.M., Kelly, K.R. and Boore, D.M. [1974] Accuracy of finite-difference modeling of the acoustic wave equation. *GEOPHYSICS*, **39**(6), 834–842.

Chang, W. and McMechan, G. [1990] 3D acoustic prestack reverse-time migration. *Geophysical Prospecting*, **38(7)**, 737–755.

Etienne, V., Tonellot, T., Malas, T., Ltaief, H., Kortas, S., Thierry, P. and Keyes, D. [2017] High-Performance Seismic Modeling with Finite-Difference Using Spatial and Temporal Cache Blocking. *3rd EAGE Workshop High Performance Computing for Upstream.*

Komatitsch, D. and Martin, R. [2007] An Unsplit Convolutional Perfectly Matched Layer Improved at Grazing Incidence for the Seismic Wave Equation. *GEOPHYSICS*, **72**(5), SM155–SM167.

Malas, T., Hager, G., Ltaief, H. and Keyes, D.E. [2017] Multidimensional Intratile Parallelization for Memory-Starved Stencil Computations. *ACM Trans. Parallel Comput.*, **4**(3), 12:1–12:32.

Malas, T., Hager, G., Ltaief, H., Stengel, H., Wellein, G. and Keyes, D. [2015] Multicore Optimized Wavefront Diamond Blocking for Optimizing Stencil Updates. *SIAM Journal on Scientific Computing*, **37**(4), 439–464.

Virieux, J. and Operto, S. [2009] An overview of full-waveform inversion in exploration geophysics.

Geophysics, **76(6)**, WCC1–WCC26.

Zhang, W. and Luo, J. [2013] Full-waveform Velocity Inversion Based on the Acoustic Wave Equation. *American Journal of Computational Mathematics*, **03**, 13–20.