Performance Optimization of Geophysics Stencils on Multicore Architectures: A Machine Learning Approach

Víctor Martínez[1] and Philippe O. A. Navaux[1]

Informatics Institute, UFRGS, Brazil
{victor.martinez,navaux}@inf.ufrgs.br

Abstract. Wave modeling is a crucial tool in geophysics (for efficient strong motion analysis, risk mitigation and oil & gas exploration). Due to its simplicity and numerical efficiency, the finite-difference method is one of the standard techniques implemented to solve these models. This kind of applications is known as stencils because they consist in a pattern that replicates the same computation on a multi-dimensional domain. The performance optimization of stencil computations is a challenge and strongly depends on the underlying architecture. In this context, this work was directed toward a twofold aim. In this sense, we have led our research on multicore architectures and we have analyzed the implementation of three numerical kernels (3D heat transfer model, seismic wave and acoustic wave propagation models). We have also considered two well-known implementations (naïve, and space blocking); thus, we proposed a Machine Learning-based approach to evaluate, to predict, and to improve the performance. Our proposed model achieves a prediction of ∼99% in several scenarios.

Keywords: HPC · Machine Learning · Multicore · Stencil Computations · Performance improvement.

1 Introduction

The behavior of scientific applications related to High Performance Computing (HPC) depends on many factors (non-uniform memory access, vectorization, compiler optimizations, memory policies, scheduling algorithms, etc.) that may severely influence its performance. It leads to progressively redesign the scientific applications to obtain the best performance for a specific architecture with a specific programming model [4, 28].

An example of HPC applications are the stencil-based applications (a nearest-neighbor pattern replicated in multidimensional data), which are used to solve many problems related to partial differential equations (PDE), the heart of many problems in areas as diverse as electromagnetics, fluid dynamics or geophysics; and the finite-difference method (FDM) is a standard technique implemented to solve these equations [29, 10, 31]. In addition to the challenging of geophysical

and mathematical problems behind seismic and acoustic wave propagation modeling, one major challenge is to leverage the various levels of parallelism involved. HPC is actually facing key challenges on both the hardware and the software sides. At hardware level, the machines are reaching several millions of cores and the scalability of the applications will become a tremendous bottleneck. As reported in several recent research papers [33, 37, 3], various geophysical applications show the variability of scaling up to several tens of thousands of cores. At software level, several parameters such as domain decomposition, compiler flags, scheduling or load balancing algorithms are considered to achieve the best performance.

Application tuning represents one methodology to improve the performance on HPC architectures. In this case, several parameters such as machine architecture, domain decomposition, compiler flags, scheduling or load balancing algorithms are considered to achieve the best performance. Unfortunately, these approaches lead to the exploration of a huge set of parameters and finding the optimal value for each parameter requires to search on a large configuration set. At this point, Machine Learning (ML) algorithms have been used on HPC systems under different situations and for various workloads such as threads mapping and memory accesses [5], I/O scheduling [2, 19], or performance improvement of stencil applications [17, 11, 6, 36, 27]. Building a suitable ML-based performance model for geophysics numerical kernels remains a challenge.

In this context, our research focuses on performance improvement of classical geophysics numerical stencils that lie at the heart of earthquake modeling, and 3D underground imaging for the oil and gas industry. Many works have been guided in two alternatives: improvement of architectural features or improvement of algorithms and implementations in specific programming models; the first alternative is out of the scope of this research and second alternative limits the performance for each implementation on the underlying architecture. Thus, we research into a third alternative that has been recently used: how to tune de application by finding an optimal input set from a configuration set of runtime parameters. This paper is organized as follows: Section 2 describes the numerical background for the geophysics kernels; Section 3 presents a ML-based model to predict and to optimize the performance on multicore architectures; Section 4 presents the related works that combine stencil computations and algorithmic optimizations; and finally, the Section 5 concludes this document, and presents the perspectives.

2 Stencil Applications

In this section, we present the numerical models analyzed in this reasearch. Because of its simplicity, the Finite-Difference Method (FDM) is widely used to design the geophysics models, when discretizing Partial Differential Equations (PDE). From the numerical analysis point of view, the FDM computational procedure consists in using the neighboring points in horizontal, vertical or diagonal directions to calculate the current point. In the case of a 3D Cartesian grid, the

computational procedure consists in using the neighboring points in the north-south, east-west and forward-backward directions to evaluate the current grid point. The stencil sweep can be expressed as iterative time domain (represented by the first loop controlled by n_times variable), and a triply nested parallel loop presented in Algorithm 1. The algorithm then moves to the next point applying the same stencil computation until the entire spatial grid have been traversed, and the time domain is completed. The number of points used in each direction depends on the order of the approximation and is of great importance for the overall performance.

Algorithm 1: Pseudocode for stencil algorithms

```
1: for t = 1 to n_times do
2:   compute in parallel
3:   for i = 1 to SIZE_X_direction do
4:     for j = 1 to SIZE_Y_direction do
5:       for k = 1 to SIZE_Z_direction do
6:         compute stencil(3D tile)
7:       end for
8:     end for
9:   end for
10: end for
```

2.1 7-point Jacobi stencil

The 7-point Jacobi stencil is a reference example of numerical kernel used in various context in order to evaluate the impact of advanced reformulation or the impact of the underlying architecture. Known to be severely memory-bound, this numerical kernel can be described as a proxy of complex stencils like those corresponding to geophysical applications. A review can be found in [11] for instance. This stencil model also corresponds to the standard discretization of the elliptic 3D Heat equation. Calculation of this numerical equation needs 7 values, one from current point plus six from neighbor points (one previous and one next on 3D axes). Figure 1 illustrates the neighbor points of stencil.

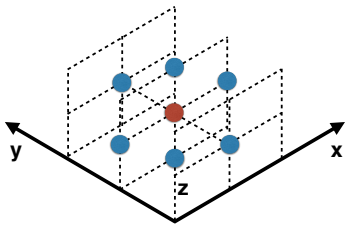

Fig. 1. Size of 7-point Jacobi stencil and its neighbor points [31].

2.2 Seismic wave propagation stencil

Evaluation of damages occurred during strong ground motion is critical for urban planning. The seismic wave equation waves radiated from an earthquake are often simulated under the assumption of an elastic medium although the waves attenuate due to some anelasticity. Due to its simplicity, the FDM is also used to compute the propagation of seismic waves. It considers a 3D isotropic elastic medium, the numerical model of the seismic wave equation is detailed in [39, 29, 22] and relies on the classical 4-th order in space and second-order in time approximation. The numerical scheme is used to compute the velocity (V_x, V_y, V_z) and the stress (σ_{xx}, σ_{yy}, σ_{zz}, σ_{xy}, σ_{xz}, σ_{yz}) components. In this case, the stencil needs 13 values, one from current point plus twelve from neighbor points (two previous and two next on 3D axes). The governing equations associated with the three-dimensional modeling of seismic wave propagation in elastic media are implemented by finite-difference discretization for *x86* cores and for GPU platforms corresponding to *Ondes3D* application developed by the *French Geological Survey (BRGM)*. Figure 2 illustrates the neighbor points of stencil.

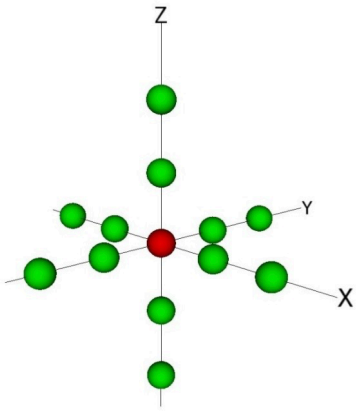

Fig. 2. Size of seismic stencil to calculate velocity and stress components [26].

2.3 Acoustic wave propagation stencil

Acoustic wave propagation approximation is the current backbone for seismic imaging tools. It has been extensively applied for imaging potential oil and gas reservoirs beneath salt domes. The problem to simulate the propagation of a single wavelet over time is solved by the isotropic acoustic wave propagation and the isotropic acoustic wave propagation with variable density under Dirichlet

boundary conditions over a finite three-dimensional rectangular domain, the numerical solution is detailed in [38], *Petrobras*, the leading Brazilian oil company, provides a standalone mini-app of the numerical method. The code was written in standard C and leverage from OpenMP directives for shared-memory parallelism. But Indeed, the parallelization strategy relies on the decomposition of the three-dimensional domain based on OpenMP loop features. Figure 3 illustrates the neighbors point of stencil.

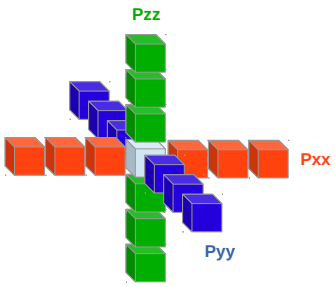

Fig. 3. Size of acoustic wave propagation stencil and its neighbor points [38].

3 Performance Improvement on Multicore Architectures

Nowadays, companies have turned to offer parallel machines, these architectures capitalize on smaller processors several processing units to provide multiple executions of instructions in the same cycle. The performance improvement is achieved by replicating the processing units, adding additional processing instructions, improving communication between the cores, and the calculations are solved simultaneously, in parallel.

3.1 Standard implementations of numerical stencil

In this section, we present the implementation of stencil algorithms for multicore platforms, and we will analyze the performance on common HPC architectures. Classical implementations on multicore architecture are related to how the threads solve each point on the space and time domain. Figure 4 shows the solution space of each algorithm. Each color represents a thread when solving one point or a set of points.

– **Naive:** On shared-memory architectures, a popular way to extract the parallelism for such applications is to exploit the triple nested loops coming from the spatial dimensions of the problem under study. This strategy allows straightforward benefits of OpenMP directives [14].

– **Space Tiling:** The second algorithm uses the cache blocking technique. In this case, the main idea is to exploit the inherent data reuse available in the triple nested loop of the kernel by ensuring that data remains in cache across multiple uses, this is achieved by creating blocks to improve data locality. The advantage of Space tiling algorithm is to improve the computational intensity by keeping a relatively small amount of data in cache memory and by performing many more floating-point operations on them. [15].

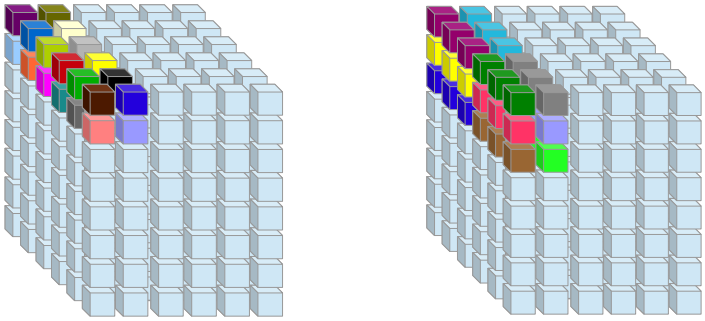

Fig. 4. Representation of points solution by threads, for naive (one point by thread) and space tiling (a group of points by thread) algorithms.

The analysis of these algorithm implementations was presented in [21]. We studied the influence of several configurations and the implemented algorithms on the performance of stencil computations. We observed that space tiling algorithm may present poor scalability in several scenarios.

3.2 General model for performance prediction

The proposed model is based on Support Vector Machine (SVM) method, which is a kernel-based and supervised approach proposed in [7] and it was extended to regression problems where support vectors are represented by kernel functions [12]. The *General Model for Performance Prediction of Geophysics Stencils based on ML (Golem)* is built on top of three consecutive layers, where the output values of a layer are used as the input values of the next layer. Figure 5 shows the flowchart of this strategy. The input layer contains the runtime configuration parameters from the input vector. The hidden layer contains a set of SVMs and takes the values from input vector to simulate the available hardware counters on the HPC architecture. Finally, the output layer contains another set of SVMs and takes each value from hardware counters layer to obtain the corresponding predicted performance (GFLOPS, and execution time).

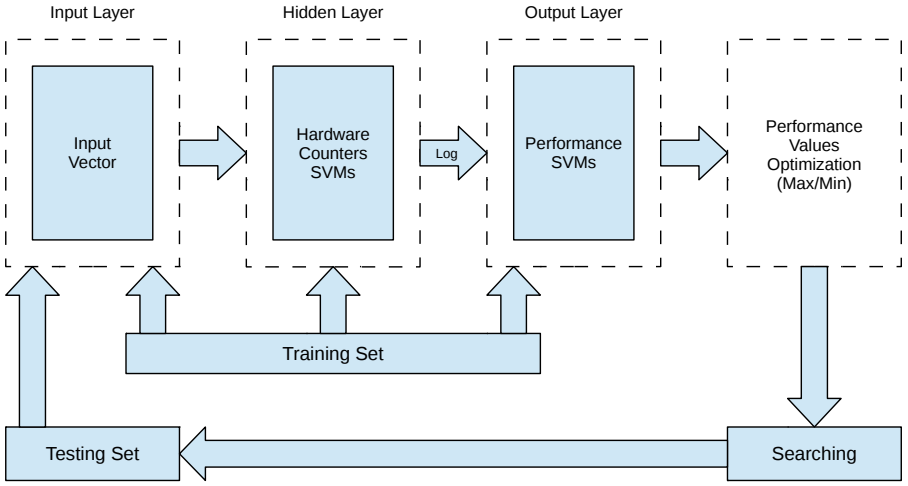

Fig. 5. Flowchart of General Model for Performance Prediction of Geophysics Stencils based on Machine Learning (Golem).

We used a set of available parameters in OpenMP defined by `OMP_NUM_THREADS` and `OMP_SCHEDULE` variables, and measures of hardware counter for training stage were taken by PAPI library [30]. We have been implemented two prediction models for multicore architectures.

The first *Golem* was implemented on two the Intel Xeon multicore architectures (E5-2650, and E5-4650) and was presented in [23]. We consider as input variables the number of threads, the loop scheduling policy (static or dynamic), and the chunk size (which defines how many loop iterations will be assigned to each thread at a time) [18]. Moreover, for the space tiling algorithm, we considered the size blocks in the X and Y domains in the blocking implementation. For the hardware counter layer, We considered as the most relevant ones the L3 total cache misses (PAPI_L3_TCM event), the data translation lookaside buffer misses (`PAPI_TLB_DM` event), and the total cycles (`PAPI_TOT_CYC` event). The output vector, which uses billions of floating-point operations per second (GFLOPS) and execution time as performance characterization metrics.

The second *Golem* was implemented on the Intel Xeon Phi 7520 architecture, and was presented in [24, 25]. The input layer includes the number of threads, the loop scheduling policy (static, dynamic or guided), the chunk size, and the memory mode (cache or flat) [35]. The hardware counter layer includes the total of L2 cache misses (measured by `PAPI_L2_TCM event`), and total of cycles (measured by `PAPI_TOT_CYC` event). The output layer includes the total elapsed time to solve the geophysical problem. In this work, we present the results of performance improvement for the second *Golem*.

Training and validation sets: We used two separated experiment sets for training and testing stages. The training set is used for input, hidden and output layers. Since the model has been trained we use an input testing set to predict the performance and we compare the output performance between predicted and actual values to obtain the regression accuracy. Finally, we search into these predicted values the maximum of GFLOPS, or the minimum of time, and its corresponding input configuration to optimize the execution time. We created a training set by randomly selecting a subset from the configuration parameters explained in previos section. Then, for each experiment, we measured the hardware counters (L2 cache misses, and total cycles) and performance (execution time). The random testing set was used since all SVMs in both the hidden and the output layers are trained to predict the execution time values. Table 1 presents the total number of experiments that were performed to obtain the training and validation sets.

Table 1. Number of experiments

	Training	Testing	**Total**
Jacobi	334	3007	**3341**
Seismic	346	3122	**3468**
Acoustic	335	3021	**3356**

Analysis of variance and hardware counters behavior: Before training stage and in order to refine the results, we applied the Analysis of Variance (ANOVA) statistical model to analyze if variables for hardware counters and performance measures have different populations affected by the input values. In this analysis, the execution time, the number of L2 cache misses, and the number of cycles were used as population variables and factors are defined by all values in vector input (the number of threads, the scheduling policy, the chunk size and memory mode). The results of *p-value* for the 7-point Jacobi, seismic and acoustic wave propagation stencils are presented in Table 2.

General considerations can be observed with results from Table 2. First, the memory mode is a factor with no statistical significance in most of variables and stencils, only for the acoustic stencil, the memory mode produces statistical significance in execution time and cycles variables. It could be explained because the code optimizations made for this stencil [34]; the Jacobi and seismic stencils have been implemented with no code optimization for the Xeon Phi architecture. Second, the Jacobi stencil has statistical significance in all variables (execution time, L2 cache misses and cycles) for the other factors. Third, variables for seismic stencil only has statistical significance by threads counting. Fourth, the acoustic stencil has statistical significance for almost all variables, except for execution time by thread counting.

Table 2. *p-value* of one-way ANOVA for the manycore architecture.

	Execution time			*L2 cache misses*			*Cycles*		
	Jacobi	*Seismic*	*Acoustic*	Jacobi	Seismic	Acoustic	*Jacobi*	*Seismic*	*Acoustic*
Scheduling policy	1.26e-13	1.000	1.95e-08	<2e-16	0.992	<2e-16	<2e-16	1.000	<2e-16
Chunk size	<2e-16	0.949	<2e-16	<2e-16	0.927	<2e-16	<2e-16	0.956	<2e-16
Num. of threads	<2e-16	<2e-16	0.444	1.82e-10	<2e-16	0.000757	<2e-16	<2e-16	2.80e-07
Memory mode	0.77	0.949	<2e-16	0.132	0.919	0.424529	0.969	0.915	5.81e-05

We calculate a two-way ANOVA to determine if combined variables affect the seismic stencil performance. Table 3 shows the results of two-way ANOVA that combining scheduling and thread counting, in the seismic stencil, rejects the hypothesis, and the variables have the statistical difference if the two factors are combined. But, when we combine chunk with another factor, we still don't find statistical significance. Then, we can summarize that most of the factors produce statistical significance into selected variables, except the chunk size, because of the analysis of variance on many-core architectures.

Table 3. *p-value* of two-way ANOVA for the seismic wave kernel.

	Execution time	*L2 cache misses*	*Cycles*
Scheduling policy:Chunk	1.000	0.997	1.000
Scheduling policy:Num. of threads	<2e-16	<2e-16	<2e-16
Chunk:Num. of threads	0.992	0.956	0.988

Performance optimization: Since the goal is to obtain the best performance, output values form the trained model were compared with measurements from all actual data. Figure 6 present results of this comparison. Blue bars represent the normalized output of the predicted best performance from the ML-based model (minimum execution time) whereas red bars represent the normalized best performance from actual data. Perfect fit of best performance is obtained when the predicted values are the same (or very close) to actual best performance values.

For Jacobi stencil, the model achieves the best performance with accuracy of 98.88%. While for the Seismic and the Acoustic kernel, the best execution time was outperformed in 112.95% and 100.59% respectively. The speedup of best performance was $\times$49.76, $\times$35.53 and $\times$39.83 for 7-point Jacobi, seismic and acoustic stencils respectively, compared to worst performance.

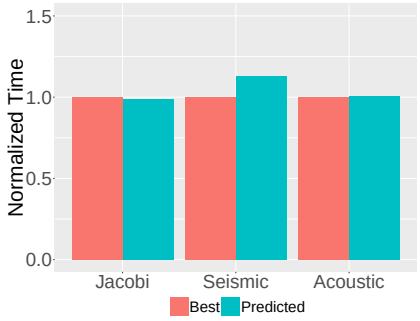

Fig. 6. Normalized performance comparison between predicted results from the ML-algorithm and results from the best performance experiments on Xeon Phi Architecture.

4 Related Works

In this section, we present the state of art from several works that are related to optimization of stencil computations. A large scientific literature has been devoted to the adaptation of stencil algorithms on HPC architectures. It involves from hardware implementations, low-level optimization, compiler flags, application tuning, programming of new algorithms, ML models to improve the performance of these applications, and task-based implementations of numerical stencils.

In [20], the authors combine the multicore temporal blocking and a diamond tiling to arrive reduce the memory pressure, the results show performance advantages in bandwidth-starved situations.

In [13] the authors also implement stencil algorithms focused on cache improvement by spacetime blocking. The trend for these HPC applications is to pay a higher cost in order to optimize the overall performance. In this work is explained the Naive and Space tiling algorithm implementations analyzed in our research.

Application tuning represents a classical methodology to improve the performance on multicore architectures. Finding the optimal value for each parameter requires to search on a large set of configurations and several heuristics or frameworks have been proposed to speed up the process of finding the best configuration for scientific applications. Unfortunately, application tuning leads to the exploration of a huge set of parameters, thus limiting its interest on complex platforms. In terms of regression models and performance prediction applied to multicore architectures, the impact of different optimizations is difficult to predict due to the influence of load imbalance, synchronization overhead, and cache locality.

In [8], the authors predict the performance behavior of stencil computations by using a model based on cache misses and prefetching. They create one con-

figuration vector and one performance vector and use [1] to obtain auto-tuned best configuration.

In [32], the authors present a performance study for stencil computations on the Intel Nehalem multicore architecture, they model the overall performance of differently optimized code based on the impact of optimizations on individual architectural components measured by hardware counters, and apply regression analysis to a large collection of empirical data to derive the model and verify the precision of the approach.

Because ML is a methodology for optimization that could be applied to find patterns on a large set of input parameters, recent works use this approach to improve the performance and to build regression models of HPC applications. Many of these works use cache-related metrics.

In [16] the authors apply ML techniques to explore stencil configurations (code transformations, compiler flags, architectural features and optimization parameters). Their approach is able to select a suitable configuration that gives the best execution time and energy consumption.

In [40] the authors propose a dynamic scheduling policy based on a regression model to ensure that is capable of responding to the changing behaviors of threads during execution.

In [9], the authors improve the performance of stencil computations by using a model based on hardware counter behavior and ML. They have included several features in the model such as multi and many-core support, hardware prefetching modeling, cache interference and capacity misses and other optimization techniques such as spatial blocking and Semi-stencil. If we compare our proposed ML model with this work we have obtained a better accuracy of the prediction results.

5 Conclusion and Perspectives

This research was addressed on the hypothesis that performance optimization of geophysics applications can be done by searching an optimal input configuration set, at runtime. In this sense, our work was focused on developing a model based on an ML approach to finding the input configuration, on multicore architectures, and researching into new programming model implementations to exploit all the computing resources on heterogeneous architectures.

We presented that the ML-based model can be adapted according to the algorithm implementations and the architectural features. Since the prediction model have been trained, it can be used to find the input configuration set to reach the optimal performance. The results of the proposed ML model proved that performance prediction of geophysics numerical kernels can be done by building a regression model with high accuracy ($\sim$99%).

This research can be extended in several ways, and we delight three possibilities as follows:

– **Exploring on new input configurations.** We proposed an improvement of shared-memory programming by a performance prediction model. We be-

lieve that many parameters can be included as input variables of the ML-based model for performance prediction. These parameters could be related to programming models (i.e., MPI-based, over-decomposition in virtual processes), compiler options (i.e., optimization flags, SIMD vectorization), or architectural features (i.e., FPGAs, embedded systems).
– **Developing a new model based on unsupervised ML algorithms.** Principal limitation of the proposed ML-based model is related to the time consumed in training and testing stages because we used a supervised method. We think that an unsupervised-based model would find the input set by auto-tuning techniques that converge to optimal performance. In this context, we may address the research towards clustering methods, genetic algorithms, or neural networks as self-organizing maps.

Acknowledgments

This work has been granted by *Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES)*, *Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq)*, *Fundação de Amparo à Pesquisa do Estado do Rio Grande do Sul (FAPERGS)*, and *Petrobras* company. This research was funded from the *EU H2020 Programme* and from *MCTI/RNP-Brazil* under the *High Performance Computing for Energy Project (HPC4E)*, grant agreement 689772, and the *Iberian-American Network for High Performance Computing (RICAP)*, partially funded by the *Ibero-American Program of Science and Technology for Development (CYTED)*, Ref. 517RT0529. The authors also want to thank Dr. *Fabrice Dupros* for providing the 7-point Jacobi and seismic wave stencils, and Dr. *Jairo Panetta* for providing the acoustic wave stencil implementations.

References

1. Bach, F.R., Jordan, M.I.: Kernel independent component analysis. J. Mach. Learn. Res. **3**, 1–48 (Mar 2003)
2. Boito, F.Z., Kassick, R.V., Navaux, P.O.A., Denneulin, Y.: Automatic I/O scheduling algorithm selection for parallel file systems. Concurrency and Computation: Practice and Experience **28**(8), 2457–2472 (2016), cpe.3606
3. Breuer, A., Heinecke, A., Bader, M.: Petascale local time stepping for the ADER-DG finite element method. In: 2016 IEEE International Parallel and Distributed Processing Symposium, IPDPS 2016, Chicago, IL, USA, May 23-27, 2016. pp. 854–863 (2016)
4. Buchty, R., Heuveline, V., Karl, W., Weiss, J.P.: A survey on hardware-aware and heterogeneous computing on multicore processors and accelerators. Concurrency and Computation: Practice and Experience **24**(7), 663–675 (2012)
5. Castro, M., Góes, L.F.W., Méhaut, J.F.: Adaptive thread mapping strategies for transactional memory applications. Journal of Parallel and Distributed Computing **74**(9), 2845 – 2859 (2014)
6. Christen, M., Schenk, O., Burkhart, H.: Automatic code generation and tuning for stencil kernels on modern shared memory architectures. Comput. Sci. **26**(3-4), 205–210 (Jun 2011)

7. Cortes, C., Vapnik, V.: Support-vector networks. Machine Learning **20**(3), 273–297 (1995)
8. de la Cruz, R., Araya-Polo, M.: Towards a multi-level cache performance model for 3d stencil computation. Procedia Computer Science **4**, 2146 – 2155 (2011)
9. de la Cruz, R., Araya-Polo, M.: Modeling Stencil Computations on Modern HPC Architectures, pp. 149–171. Springer International Publishing, Cham (2015)
10. Datta, K., Murphy, M., Volkov, V., Williams, S., Carter, J., Oliker, L., Patterson, D., Shalf, J., Yelick, K.: Stencil computation optimization and auto-tuning on state-of-the-art multicore architectures. In: Proceedings of the 2008 ACM/IEEE Conference on Supercomputing. pp. 4:1–4:12. SC '08, IEEE Press, Piscataway, NJ, USA (2008)
11. Datta, K., Williams, S.W., Volkov, V., Carter, J., Oliker, L., Shalf, J., Yelick, K.: Auto-Tuning Stencil Computations on Multicore and Accelerators. CRC Press, Taylor & Francis Group (2010)
12. Drucker, H., Burges, C.J.C., Kaufman, L., Smola, A., Vapnik, V.: Support vector regression machines. In: Advances in Neural Information Processing Systems 9. pp. 155–161. MIT Press (1997)
13. Dupros, F., Boulahya, F., Aochi, H., Thierry, P.: Communication-avoiding seismic numerical kernels on multicore processors. In: High Performance Computing and Communications (HPCC), 2015 IEEE 7th International Symposium on Cyberspace Safety and Security (CSS), 2015 IEEE 12th International Conferen on Embedded Software and Systems (ICESS), 2015 IEEE 17th International Conference on. pp. 330–335 (Aug 2015)
14. Dupros, F., Aochi, H., Ducellier, A., Komatitsch, D., Roman, J.: Exploiting Intensive Multithreading for the Efficient Simulation of 3D Seismic Wave Propagation. In: Proceedings of the 11th IEEE CSE'08, International Conference on Computational Science and Engineering. pp. 253–260. São Paulo, Brazil (July 2008)
15. Dupros, F., Do, H., Aochi, H.: On scalability issues of the elastodynamics equations on multicore platforms. In: Proceedings of the International Conference on Computational Science, ICCS 2013, Barcelona, Spain, 5-7 June, 2013. pp. 1226–1234 (2013)
16. Ganapathi, A., Datta, K., Fox, A., Patterson, D.: A case for machine learning to optimize multicore performance. In: Proceedings of the First USENIX Conference on Hot Topics in Parallelism. pp. 1–1. HotPar'09, USENIX Association, Berkeley, CA, USA (2009)
17. Ganapathi, A.S.: Predicting and Optimizing System Utilization and Performance via Statistical Machine Learning. Ph.D. thesis, EECS Department, University of California, Berkeley (Dec 2009)
18. Intel: OpenMP* Loop Scheduling. https://software.intel.com/en-us/articles/openmp-loop-scheduling (2014), [Online; accessed 11-jan-2018]
19. Li, Y., Chang, K., Bel, O., Miller, E.L., Long, D.D.E.: Capes: Unsupervised storage performance tuning using neural network-based deep reinforcement learning. In: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis. pp. 42:1–42:14. SC '17, ACM, New York, NY, USA (2017)
20. Malas, T.M., Hager, G., Ltaief, H., Stengel, H., Wellein, G., Keyes, D.E.: Multicore-optimized wavefront diamond blocking for optimizing stencil updates. SIAM J. Scientific Computing **37**(4) (2015)
21. Martinez, V., Dupros, F., Castro, M., Aochi, H., Navaux, P.O.A.: Stencil-based applications tuning for multi-core. In: Latin American High Performance Computing Conference (CARLA 2016). pp. 1–15 (Sep 2016), oral presentation

22. Martinez, V., Michea, D., Dupros, F., Aumage, O., Thibault, S., Aochi, H., Navaux, P.O.A.: Towards seismic wave modeling on heterogeneous many-core architectures using task-based runtime system. In: Computer Architecture and High Performance Computing (SBAC-PAD), 2015 27th International Symposium on. pp. 1–8 (Oct 2015)
23. Martinez, V., Dupros, F., Castro, M., Navaux, P.: Performance improvement of stencil computations for multi-core architectures based on machine learning. Procedia Computer Science **108**, 305 – 314 (2017), international Conference on Computational Science, ICCS 2017, 12-14 June 2017, Zurich, Switzerland
24. Martínez, V., Serpa, M., Dupros, F., Padoin, E.L., Navaux, P.: Performance prediction of acoustic wave numerical kernel on intel xeon phi processor. In: Mocskos, E., Nesmachnow, S. (eds.) High Performance Computing. pp. 101–110. Springer International Publishing, Cham (2018)
25. Martinez, V., Serpa, M.S., Navaux, P.O.A., Padoin, E.L., Panetta, J.: Performance improvement of stencil computations for multi-core architectures based on machine learning. In: ENERGY 2018 : The Eighth International Conference on Smart Grids, Green Communications and IT Energy-aware Technologies. pp. 76 – 81. IARIA (2018)
26. Michéa, D., Komatitsch, D.: Accelerating a 3D finite-difference wave propagation code using GPU graphics cards. Geophysical Journal International **182**(1), 389–402 (2010)
27. Mijakovic, R., Firbach, M., Gerndt, M.: An architecture for flexible auto-tuning: The periscope tuning framework 2.0. In: International Conference on Green High Performance Computing (ICGHPC). pp. 1–9 (Feb 2016)
28. Mittal, S., Vetter, J.S.: A survey of cpu-gpu heterogeneous computing techniques. ACM Comput. Surv. **47**(4), 69:1–69:35 (Jul 2015)
29. Moczo, P., Robertsson, J., Eisner, L.: The finite-difference time-domain method for modeling of seismic wave propagation. In: Advances in Wave Propagation in Heterogeneous Media, Advances in Geophysics, vol. 48, chap. 8, pp. 421–516. Elsevier - Academic Press (2007)
30. Mucci, P.J., Browne, S., Deane, C., Ho, G.: Papi: A portable interface to hardware performance counters. In: In Proceedings of the Department of Defense HPCMP Users Group Conference. pp. 7–10 (1999)
31. Nguyen, A., Satish, N., Chhugani, J., Kim, C., Dubey, P.: 3.5-d blocking optimization for stencil computations on modern cpus and gpus. In: 2010 ACM/IEEE International Conference for High Performance Computing, Networking, Storage and Analysis. pp. 1–13 (Nov 2010)
32. Rahman, S.M.F., Yi, Q., Qasem, A.: Understanding stencil code performance on multicore architectures. In: Proceedings of the 8th ACM International Conference on Computing Frontiers. pp. 30:1–30:10. CF '11, ACM, New York, NY, USA (2011)
33. Roten, D., Cui, Y., Olsen, K.B., Day, S.M., Withers, K., Savran, W.H., Wang, P., Mu, D.: High-frequency nonlinear earthquake simulations on petascale heterogeneous supercomputers. In: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis, SC 2016, Salt Lake City, UT, USA, November 13-18, 2016. pp. 957–968 (2016)
34. Serpa, M.S., Cruz, E.H.M., Diener, M., Krause, A.M., Farrs, A., Rosas, C., Panetta, J., Hanzich, M., Navaux, P.O.A.: Strategies to improve the performance of a geophysics model for different manycore systems. In: 2017 International Symposium on Computer Architecture and High Performance Computing Workshops (SBAC-PADW). pp. 49–54 (Oct 2017)

35. Sodani, A., Gramunt, R., Corbal, J., Kim, H.S., Vinod, K., Chinthamani, S., Hutsell, S., Agarwal, R., Liu, Y.C.: Knights landing: Second-generation intel xeon phi product. IEEE Micro **36**(2), 34–46 (Mar 2016)
36. Tang, Y., Chowdhury, R.A., Kuszmaul, B.C., Luk, C.K., Leiserson, C.E.: The pochoir stencil compiler. In: ACM Symposium on Parallelism in Algorithms and Architectures. pp. 117–128. SPAA '11, ACM, New York, NY, USA (2011)
37. Tsuboi, S., Ando, K., Miyoshi, T., Peter, D., Komatitsch, D., Tromp, J.: A 1.8 trillion degrees-of-freedom, 1.24 petaflops global seismic wave simulation on the K computer. IJHPCA **30**(4), 411–422 (2016)
38. Vilela, R.F.: Perfilagem do problema de resolução da equação da onda por diferenças finitas em coprocessador xeon phi (2017), dissertação (mestrado)
39. Virieux, J.: P-SV wave propagation in heterogeneous media; velocity-stress finite-difference method. Geophysics **51**(4), 889–901 (1986)
40. Weng, L., Liu, C., Gaudiot, J.L.: Scheduling optimization in multicore multithreaded microprocessors through dynamic modeling. In: Proceedings of the ACM International Conference on Computing Frontiers. pp. 5:1–5:10. CF '13, ACM, New York, NY, USA (2013)