The Roofline Model for Oceanic Climate Applications

Italo Epicoco*†, Silvia Mocavero†, Francesca Macchia† and Giovanni Aloisio*†
*University of Salento, Lecce, Italy
Email: italo.epicoco, giovanni.aloisio @unisalento.it
†Euro-Mediterranean Center on Climate Change, Lecce, Italy
Email: silvia.mocavero, francesca.macchia@cmcc.it

***Abstract*—The present work describes the analysis and optimisation of the PELAGOS025 configuration based on the coupling of the NEMO physic component of the ocean dynamics and the BFM (Biogeochemical Flux Model), a sophisticated biogeochemical model that can simulate both pelagic and benthic processes. The methodology here followed is characterised by the performance analysis of the original parallel code, in terms of strong scalability, the definition of the bottlenecks limiting the scalability when the number of processes increases, the analysis of the features of the most computational intensive kernels through the Roofline model which provides an insightful visual performance model for multicore architectures and which allows to measure and compare the performance of one or more computational kernels run on different hardware architectures.**

***Keywords*—Roofline; performance analysis; oceanic model**

I. Introduction

Climate models simulate the complex climate system integrating one or more components, such as atmosphere, ocean, biogeochemistry, land, ice, etc., which can interact each other, allowing to study the climate change and its impact on the environment, economy and society [1] [2].
Many simulation models have been developed several decades ago and they only partially exploit the computational power of the new generation supercomputers. They require a deep re-thinking in order to better use the available computational resources and to drastically reduce the time-to-solution [3].
There are two feasible approaches: a re-design of the model core from scratch, starting from the physics modeling up to the implementation phase, passing through the use of innovative numerical schemas and algorithms; the second approach is based on the optimization of the existing code to improve the performance taking into account a target architecture. Some works follow the first approach developing new dynamical cores able to improve the simulation accuracy at high resolution and to improve the scalability on the new generation parallel architectures. Dennis et al. designed and developed the HOMME dycore [4], while Govett et al. evaluated the NIM next-generation weather model, developed at NOAA, on GPUs [5]. Other works point to the second approach, analyzing and optimizing the model on a target architecture. In the past, we have developed a performance model for the SOR algorithm used by the NEMO oceanic model [6] in order to better exploit the MareNostrum platform at the Barcelona Supercomputing Center [7]. Reid [8] evaluated and optimized NEMO performance on the Cray XT4 HECToR by investigating the choice of grid dimensions, by examining the use of land versus ocean grid cells and also by checking for memory bandwidth problems. Porter et al. worked to optimize and to port the NEMO model on GPUs using accelerator directives with the aim of minimizing changes to the original Fortran source code [9].
The described approaches are not alternative solutions and can coexist. However, the first one implies a long-term activity and it requires an interdisciplinary effort among physicists, mathematicians, computational scientists and technology providers; the second one is a short-term operation and can be performed by the computer scientists. Obviously, the performance improvement got due to the first approach is more tangible, but the necessity to improve the time-to-solution for the climate scientists in the short-time incites to simultaneously follow both the approaches.
The present work is an example of the first approach. Generally, the optimization methodology is characterized by the following steps: (i) the performance analysis of the original parallel code, in terms of strong scalability on a target architecture; (ii) the code profiling to identify the bottlenecks limiting the scalability when the number of processes increases; (iii) the analysis of the computational issues of the most computational intensive kernels. The analysis of the single kernel allows to act on few code sections, without completely changing the whole code, and to get good benefits from little improvements. Often, a low level optimization such as the integration of optimized mathematic libraries (PETSc [10] [11] and ScaLAPACK [12]) or an optimal process binding on the resources in order to enhance the data locality and to better exploit the target architecture, can give a relevant improvement of the performance. The scalability is often limited by the computational and the memory capacity of the target architecture w.r.t. the application requirements. The analysis of this relation can be carried out using the "roofline model" [13]. More details about roofline will be provided in the section 3 with the description of the IBM cluster based on SandyBridge processors; the section 4 describes the analysis performed using the roofline on the PELAGOS025 model obtained from the coupling between NEMO and the BFM

biogeochemical model. The next section details the climate model on which the work is focused.

II. The PELAGOS oceanic model

The focus of the present work is a coupled version of the NEMO oceanic model. NEMO is a 3-dimensional ocean model used for oceanography, climate modeling as well as operational ocean forecasting. It is a finite-difference model with a regular domain decomposition and a tripolar grid with each of the poles posed on land to prevent singularities. NEMO calculates the incompressible Navier-Stokes equations on a rotating sphere. The prognostic variables are the three-dimensional velocity, temperature and salinity and the surface height. The distribution of variables is a three-dimensional Arakawa C-type grid. The code is written in Fortran 90 and parallelized with MPI using the domain decomposition approach.
NEMO is used in more than 50 research projects for both long and short-term simulations with different configurations and several spatial and time resolutions. In some configurations it is used as stand-alone model, otherwise it is coupled with other components such as the atmospheric model or biogeochemistry. The analysis performed in this paper considers the PELAGOS025 configuration based on the coupling of the NEMO physic component and the BFM (Biogeochemical Flux Model), a sophisticated biogeochemical model that can simulate both pelagic and benthic processes [14]. BFM is a numerical model, which describes the dynamics of major biogeochemical processes occurring in marine systems. It considers the cycles of nitrogen, phosphorus, silica, carbon, and oxygen in the water dissolved phase, as well as in the plankton, detritus, and benthic compartments. Plankton dynamics are parameterized by considering a number of plankton functional groups, each representing a class of taxa. From a mathematical point of view, the BFM is a set of ordinary differential equations describing the time rate of change of the concentration of a number of biogeochemically active tracers. The coupled model has been tested in a global configuration with a horizontal resolution of 0.25 degree and 50 vertical levels. It uses 57 pelagic variables.
The combination of the two models poses several computational challenges:

- high global spatial resolution with many grid points and a small numerical time step;
- large number of biogeochemical variables to be transported, in addition to the ocean model active tracers;
- computation of diagnostic output across sub-domains;
- storage of large amount of data with various time frequencies.

The execution time of the coupled model is strongly influenced by these needs, then the use of massive parallel architectures represents a pressing requirement. The model has been tested on an iDataPlex cluster equipped with Intel SandyBridge processors (named ATHENA), located at the CMCC Supercomputing Center in Lecce. The first step has been the choice of the domain decomposition along the two horizontal directions when the number of total MPI tasks increases. The best strategy is to select the two dimensions allowing the subdomain as much square as possible. This way, the communication overhead is minimum. However, adding the biogeochemical component, the number of the ocean points for each subdomain becomes a crucial factor, since BFM unlike NEMO performs computation only on these points. A preprocessing tool can be used to establish the best configuration which minimizes the number of ocean points of the subdomain with the biggest number of them. Each experiment has been repeated 5 times with 30 total runs. For each run we simulated one day with a temporal discretization of 1,080 seconds per time step (a total of 80 time steps for each run). The charts in figures 1, 2 and 3 show the scalability results respectively in terms of speedup, execution time and SYPD (Simulated Years Per Day), a metric for measuring the simulation throughput usually referred by the climate scientists to evaluate the model performance.
The tests were be executed up to 2,048 cores. As you can see in the figure 2, the execution time on 2,048 cores increases w.r.t. the run on 1,728 cores. For this reason we have stopped the analysis to 2,048 cores.
The second step, after the analysis of scalability, is the determination of the bottlenecks, which limit it. The following section introduces the roofline model used to analyze the single kernels and its features.

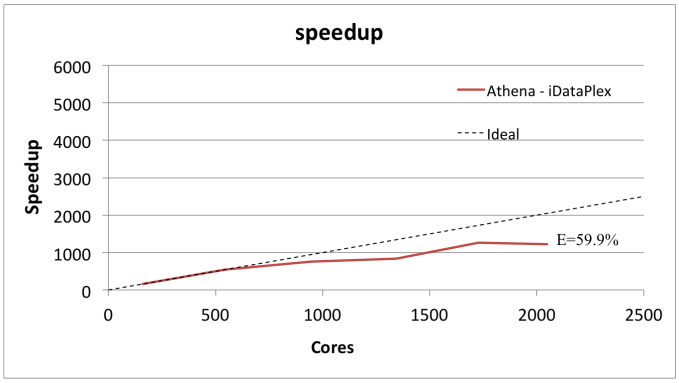

Figure 1. Scalability of PELAGOS025 configuration. Red line represents the speedup of the model on ATHENA, dash line is an indication of the ideal speedup.

III. The Roofline Model

The roofline model is a model that allows to measure and compare the performance of one or more computational kernels executed on different hardware architectures. It aims at providing an insight into the performance on many core architecture; it does not need to be a perfect performance model but just insightful. The basic idea behind the roofline model is that an application is limited by two main factors: the peak floating point operations per seconds and the maximum memory bandwidth required to execute a given number of

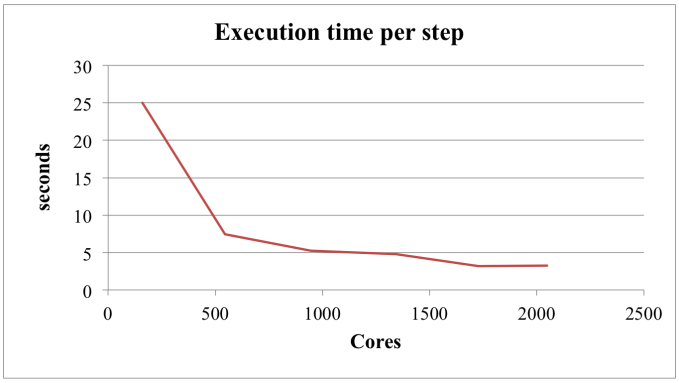

Figure 2. Scalability of PELAGOS025 configuration: execution time/time step of the model on ATHENA.

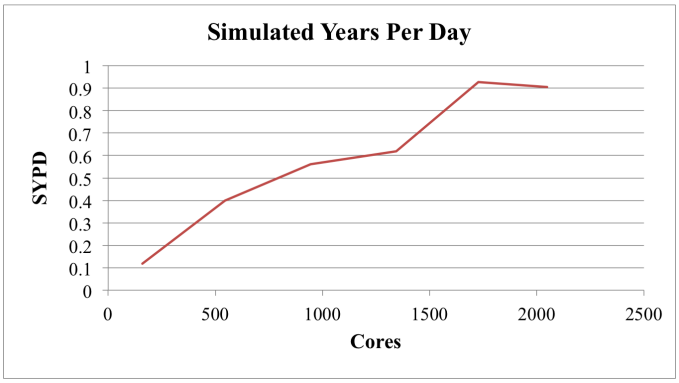

Figure 3. Scalability of PELAGOS025 configuration: Simulated Years Per Day of the model on ATHENA.

operations per seconds. Hence, to build the roofline model the following characteristics of the architecture must be taken into account:

- max Floating Point Operations per second;
- max Memory Bandwidth.

The first is the maximum number of floating point operations that a core can execute and often, for multicore chips, it is evaluated as the collective peak performance of all the cores on the chip. The second number refers to the Max Memory Bandwidth (MMB) of the architecture, i.e. the maximum speed at which you can transfer the data from/to the memory.
The other data we need are related to the computational kernel that we want to evaluate. First of all we need the total number of floating point operations performed by the kernel during a run, measured in GFlops. Another important factor to take into account is the Arithmetic Intensity (AI), which is the number of floating point operations performed per byte of memory in which the computational kernels had access. It is measured in (Floating Point Operations)/Byte and it can be considered as a measure of the density of all floating-point operations performed related to the total number of bytes of data transferred from DRAM.
The target architecture we have considered is ATHENA, a parallel cluster with 482 IBM iDataPlex dx360 M4 nodes, each one composed of:

- two Intel Xeon E5-2670 SandyBridge 2.6 GHz and 8 cores, for a total of 16 cores per node;
- 64GB of RAM (4 GB for each core);
- three cache levels, 32KB, 256KB and 20MB respectively.

The first two cache levels are local to each core, while the last one is shared by all of the 8 cores inside the processor. Thanks to the vector unit and the AVX instruction set, each core is able to perform 8 double precision floating point operations per clock cycle, expressing a computing capacity for each node given by equation 1

$$16(cores/node) * 8(Flops/core) * 2,6(GHZ) = 332.8(GFlops)/node) \quad (1)$$

The entire cluster is able to perform 160.41 TFlops. In addition, the processor supports a 2-way Simultaneous Multi Threading (SMT), which allows to run two threads simultaneously on each core; so, activating SMT, we can run 32 threads per node.
At this point we have to define the Roofline Model on ATHENA (Figure4).

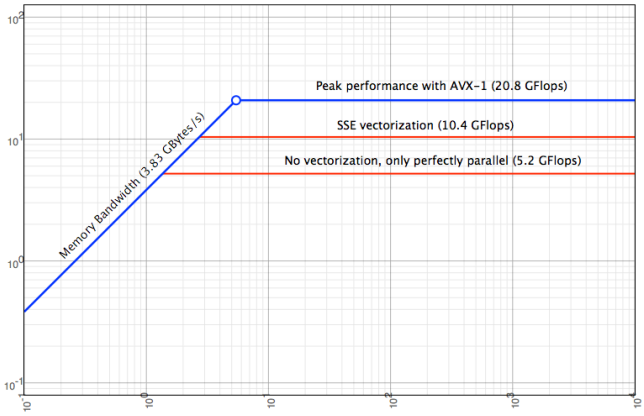

Figure 4. Roofline model with ceilings for Athena.

Peak floating-point performance can be found through the hardware specifications of the Intel Xeon E5-2670. Graphically we report three horizontal upper bounds in GFlops, related to the particular technology or instruction set used:

- **No vectorization, only perfectly parallel**: is the maximum number of GFlops reachable using a perfectly parallelized code without the use of special set of instructions dedicated to the operations of vectorization;
- **SSE (Streaming SIMD Extensions)**: is a set of instructions made available by Intel to increase the performance when the same calculation is performed on different sets of data;
- **AVX-1 (Advanced Vector eXtensions):** is a new set of instructions available with the new Intel Sandy Bridge ar-

chitecture which allows a further increase in performance achieving 20.8 GFlops.

The upper bound for memory performance can be found through the STREAM benchmark [15]. Using the EP-STREAM test we obtain a memory bandwidth of 3.82828 GBytes/s. Graphically this ceiling is represented by the line of equation 2:

$$GFlops = 3.82828(GBytes/s) * A.I. \quad (2)$$

and is upper bounded from the line representing the peak performance with AVX-1, that has equation 3:

$$y = 20.8 \quad (3)$$

So the ridge point has coordinates:

$$(5.433, 20.8)$$

IV. Kernels' Analysis

Once defined the roofline model with ceiling for ATHENA, we report the data related to the PELAGOS025 kernels. We profiled the code using *gprof* utility in order to identify the most computationally intensive ones. The test run was executed on 160 cores (6 x 29 subdomains) and carried out a simulation of 40 time steps. Figure 5 shows the output of the profiling and describes the dependences among kernels. We identify 8 kernels computationally more relevant. The data relating to the measurements are shown in TABLE I.

TABLE I
PROFILING DATA FOR THE 8 KERNELS COMPUTATIONALLY MORE RELEVANT OF PELAGOS025

kernel	% time	self s	calls	self s/call	tot. s/call
tra_ldf_iso	16.61	61.98	2120	0.029	0.029
tra_adv_muscl	16.15	60.29	2120	0.028	0.028
tra_zdf_imp	11.6	43.29	2120	0.020	0.020
extract2d	3.99	14.9	2457	0.006	0.006
mesozoodyn.	3.61	13.49	80	0.168	0.299
microzoodyn.	2.58	9.63	80	0.120	0.244
phytodyn.	2.53	9.46	120	0.078	0.157
pelglobaldyn.	2	7.46	40	0.186	0.186

To position each kernel into the roofline chart we must measure three values:

1) the number of floating point operations executed by the kernel;
2) the execution time of the kernel;
3) the amount of data transferred from/to the main memory.

The position of a kernel into the roofline chart provides an idea on how much it exploits the computational architecture and indicates what kind of actions can be performed to improve the code efficiency. The number of floating point operations measured is divided by the execution time, obtaining the ordinate value of the point relative to the specific kernel. The data bytes transferred from the memory, however, are divided by the total number of Flops to compute the AI, the abscissa of the computational kernel in the roofline chart. We used the PAPI (Performance Application Programming Interface) [16] to extract these data. While the number of floating point operations is directly reported by the native events FP_COMP_OPS_EXE, the amount of data transferred from the main memory has been derived considering the number of L3 cache misses using the event PAPI_L3_TCM multiplied by the L3 cache line.
To better evaluate the hardware counters avoiding interference between measures, we performed two runs, one to extract the cache misses value and the other one to evaluate the floating points for each kernel. The tests have been performed on 160 (6 x 29 subdomains), 944 (52 x 24 subdomains) and 2048 (122 x 23 subdomains) cores. Each test was repeated three times in order to mitigate the effect of jitter introduced by the operating system and the management services of the cluster, for a total of 18 submissions. The data obtained are reported in TABLE II.

TABLE II
DATA RELATING TO THE CONSTRUCTION OF THE ROOFLINE MODEL, TAKEN FOR EACH KERNEL UNDER CONSIDERATION AND EVALUATED FOR RUN PERFORMED ON 160, 944 AND 2048 CORES.

Kernel	Time (s)	GFlops	Mem. Access (GB)	AI
tra_ldf_iso	0.079	2.19	0.0216	7.54
	0.003	2.98	0.0008	10.42
	0.004	2.57	0.0010	11.12
tra_adv_muscl	0.048	2.57	0.0136	8.59
	0.007	2.97	0.0013	15.48
	0.002	2.57	0.0004	13.85
tra_zdf_imp	0.050	2.58	0.0233	5.25
	0.005	2.97	0.0014	11.17
	0.002	2.78	0.0005	15.05
extract2d	0.037	2.29	0.0313	2.57
	0.003	2.92	0.0040	2.05
	0.001	2.80	0.0018	1.95
mesozoodyn.	1.694	2.57	3.9516	1.02
	0.253	2.97	0.5926	1.18
	0.104	2.96	0.2358	1.21
microzoodyn.	1.691	2.55	3.7412	1.07
	0.251	2.94	0.5873	1.17
	0.108	2.97	0.2399	1.25
phytodyn.	1.076	2.34	1.9561	1.20
	0.138	2.94	0.2805	1.35
	0.059	2.97	0.1150	1.43
pelglobaldyn.	0.8125	2.57	1.8455	1.05
	0.123	2.97	0.2684	1.27
	0.055	2.93	0.1201	1.27

Plotting these data we can see that all of the kernels are below the first upper limit ("No vectorialization, only perfect parallel"), that can be achieved without using neither SSE or AVX-1 sets of instructions, but only through a perfect parallelization of the code. Since the kernel with the highest number of GFlops is *tra_ldf_iso* (2.988 GFlops peak achieved during testing on 944 and 2048 cores), we can conclude that the computing power exploited is about 14% of the total power (20.8 GFlops theoretical for ATHENA).

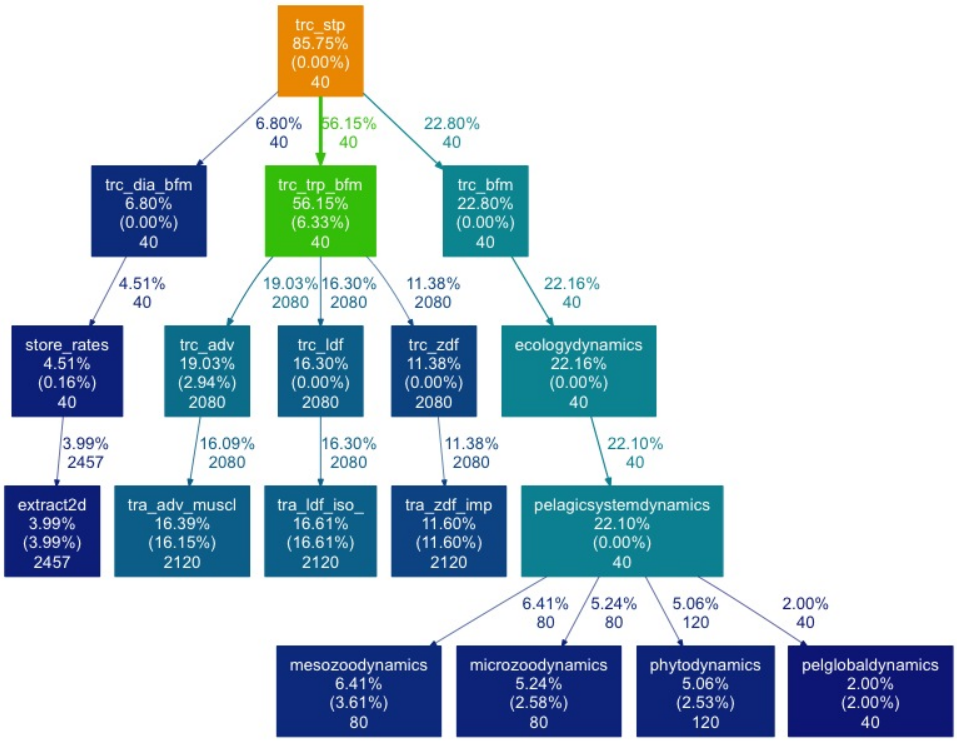

Figure 5. Profiling output and dependences among kernels for PELAGOS025.

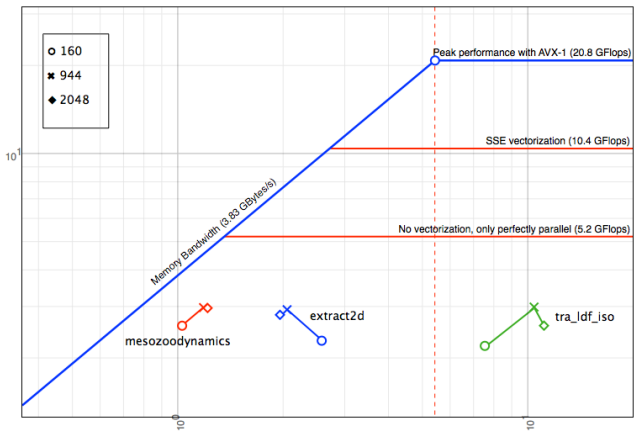

Figure 6. Roofline model for *tra_ldf_iso*, *extract2d* and *mesozoodynamics* kernels evaluated with 160, 944 and 2048 cores.

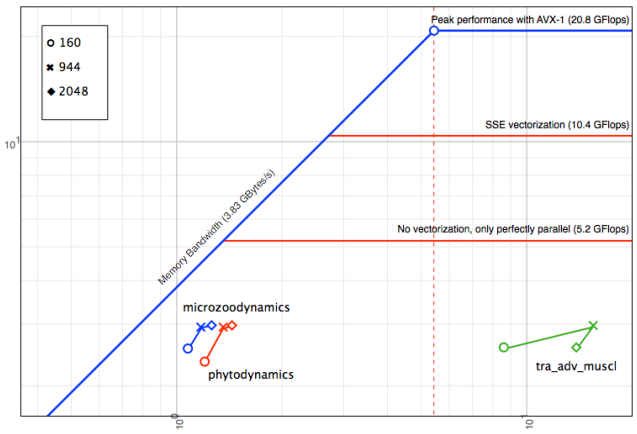

Figure 7. Roofline model for *microzoodynamics*, *tra_adv_muscl* and *phytodynamics* kernels evaluated with 160, 944 and 2048 cores.

We remind that the ridge point has coordinates (5.433, 20.8). To the right of this value we find the kernels *tra_ldf_iso*, *tra_adv_muscl* and *tra_zdf_imp*. Since these kernels are located below the line "No vectorialization, only perfect parallel", it is appropriate to increase the vectorization in order to exploit SSE or AVX-1 sets of instructions and to achieve a memory bandwidth of 20.8 GFlops. The kernels located to the left of the ridge point (*extract2d*, *mesozoodynamics*, *microzoodynamics*, *phytodynamics* and*pelglobaldynamics*), however, can not be improved beyond a certain threshold due to the memory bandwidth. Therefore we must try to increase the AI to get as close as possible the ridge point and eventually to overcome it. To this end, a redesign process of the code could be appropriate trying to improve the memory management and the load balancing.

V. Conclusion

In this paper we presented the methodology to perform the analysis of complex parallel applications such as coupled climate models. The application we have considered

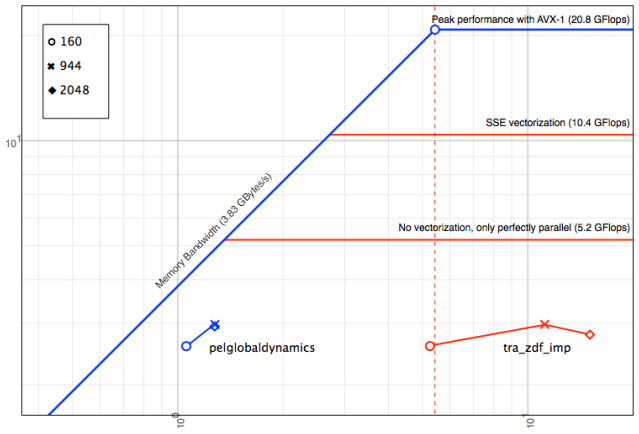

Figure 8. Roofline model for *pelglobaldynamics* and *tra_zdf_imp* kernels evaluated with 160, 944 and 2048 cores.

is a combination of two models: an ocean dynamics and a biogeochemical model. This configuration poses several computational challenges: a high global spatial resolution with many grid points and a small numerical time step; a large number of biogeochemical variables to be transported, in addition to the ocean model active tracers; the computation of diagnostic output across sub-domains; the storage of large amount of data with various time frequencies. This implies also big challenges in terms of performance optimisation. In the paper we stated that currently the model has a limit of scalability around 2048 cores and it is not suitable for exascale architectures. Moreover, considering the roofline analysis we can conclude that the model just exploits only 1% of the processor peak performance. In order to increase the performance, the level of instruction parallelism must be enhanced as well as the exploitation of the loops vectorisation. The analysis highlighted also a limit of scalability due to the load imbalance in the BFM component. A further level of optimisation can be introduced through an hybrid parallelisation based on the OpenMP+MPI paradigm in order to better exploit many core architectures.

REFERENCES

[1] UNEP, *The role of ecosystem management in climate change adaptation and disaster risk reduction*, 2009.

[2] IPCC Fifth Assessment Report (AR5), *Climate Change 2013: The Physical Science Basis*, 2013.

[3] *Scientific grand challenges: challenges in climate change science and the role of computing at the extreme scale*, Report from the Workshop Held November 6-7, 2008, Washington D.C.

[4] J.M. Dennis, J. Edwards, K.J. Evans, O. Guba, P.H. Lauritzen, A.A. Mirin, A. St-Cyr, M.A. Taylor, P.H. Worley, *CAM-SE: A scalable spectral element dynamical core for the Community Atmosphere Model*, International Journal of High Performance Computing Applications, 26-1, 74-89, ISSN 1741-2846, DOI 10.1177/1094342011428142, 2012

[5] M.W. Govett, J. Middlecoff, T. Henderson, *Running the NIM Next-Generation Weather Model on GPUs*, Proceedings of the 2010 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing, ISBN 978-0-7695-4039-9, 702-796 DOI 10.1109/CCGRID.2010.106, IEEE Computer Society, Washington, DC, USA, 2010

[6] G. Madec and the NEMO team, *NEMO ocean engine*, Note du Pole de modellisation, Institut Pierre-Simon Laplace (IPSL), France, 27, ISSN 1288-1619, 2008

[7] I. Epicoco, S. Mocavero, G. Aloisio, *The performance model for a parallel SOR algorithm using the red-black scheme*, Int. J. High Performance Systems Architecture, Vol. 4, No. 2, 2012

[8] F. J. L. Reid, *The NEMO Ocean Modelling Code: A Case Study*, EPCC, The University of Edinburgh, James Clerk Maxwell Building, Mayfield Road, Edinburgh, EH9 3JZ, UK, 2010

[9] A.R. Porter, S.M. Pickles, M. Ashworth, *Final report for the gNEMO project: porting the oceanographic model NEMO to run on many-core devices* DL Technical Reports DL-TR-2012-001. 2012 http://purl.org/net/epubs/work/62085

[10] S. Balay, K. Buschelman, W. Gropp, D. Kaushik, M. G. Knepley, L. C. McInnes, B. F. Smith, H. Zhang, *PETSc Web page*, 2001. http://www.mcs.anl.gov/petsc.

[11] S. Balay, W. D. Gropp, L. C. McInnes, B. F. Smith, *Efficient management of parallelism in object oriented numerical software libraries*. In E. Arge, A. M. Bruaset, and H. P. Langtangen, editors, Modern Software Tools in Scientific Computing, 163-202. Birkhauser Press, 1997.

[12] L. S. Blackford, J. Choi, A. Cleary, E. DAzevedo, J. Demmel, I. Dhillon, J. Dongarra, S. Hammarling, G. Henry, A. Petitet, K. Stanley, D. Walker, and R. C. Whaley, *ScaLAPACK Users Guide*. Society for Industrial and Applied Mathematics, 1997.

[13] S. Williams, A. Waterman, D. Patterson, *Roofline: An Insightful Visual Performance Model for Multicore Architectures*. Communication of the ACM, Vol. 52, No. 4, 65-76, 2009

[14] The BFM System Team. *The Biogeochemical Flux Model (BFM) Equation Description and User Manual BFM version 5* (BFMV5).

[15] McCalpin, J. *STREAM: Sustainable Memory Bandwidth in High-Performance Computers*, 1995; www.cs.virginia.edu/stream.

[16] PAPI website, http://icl.cs.utk.edu/papi/docs/index.html