Energy analysis of a 4D Variational Data Assimilation algorithm and evaluation on ARM-based HPC systems

Rossella Arcucci[1], Davide Basciano[1], Alessandro Cilardo[1], Luisa D'Amore[1], and Filippo Mantovani[2]

[1] University of Naples Federico II, Italy.
[2] Barcelona Supercomputing Center (BSC), Spain

Abstract. Driven by the emerging requirements of High Performance Computing (HPC) architectures, the main focus of this work is the interplay of computational and energetic aspects of a Four Dimensional Variational (4DVAR) Data Assimilation algorithm, based on Domain Decomposition (named DD-4DVAR). We report first results on the energy consumption of the DD-4DVAR algorithm on embedded processor and a mathematical analysis of the energy behaviour of the algorithm by assuming the architectures characteristics as variable of the model. The main objective is to capture the essential operations of the algorithm exhibiting a direct relationship with the measured energy. The experimental evaluation is carried out on a set of mini-clusters made available by the Barcelona Supercomputing Center.

Keywords: Data Assimilation, 4DVar, domain decomposition, embedded processor architectures, energy consumption

1 Introduction and Motivations

Data assimilation (DA) is an uncertainty quantification technique by which measurements and model predictions are combined to obtain an accurate representation of the state of the modeled system [5, 7]. Due to the scale of the forecasting area and the number of state variables used to describe ocean or atmosphere for climate or weather predictions, DA applications are large scale problems that should be solved in near real-time. This mandates to design and develop DA algorithms to be run by exploiting High Performance Computing (HPC) environments. During the last 20 years, parallel algorithms for DA have been investigated by a number of federal research institutes and universities. Up to now, the main efforts towards the development of parallel 4DVAR DA systems were achieved in numerical weather prediction applications, namely by the ECMWF (European Centre for Medium-Range Weather Forecasts), in Reading (UK) and by the NCAR (National Center for Atmospheric Research), in Colorado (USA). In this paper, we employ a 4DVAR algorithm described in [1, 6], named DD-4DVAR, based on a Domain Decomposition approach. In [3, 9–11] are described

The final publication is available at Springer via https://link.springer.com/chapter/10.1007/978-3-319-78054-2_4

some different approaches to take full advantage of emerging HPC architectures. In the model we employ, the parallelism is achieved by dividing the global problem into multiple local 4DVAR DA sub-problems solved across processors. The global solution is obtained by collecting the local minimums. The sub-problems are handled by a slightly modified 4DVAR algorithm, custom implemented on an ARM-based low-energy node with the aim of minimizing the overall energy-to-solution experienced by the application.
The performance and energy cost of a parallel algorithm executing on HPC systems have different trade-offs, depending on how many processors the algorithm uses, at what characteristics these processors have, and the structure of the algorithm. Due to the interest of the HPC community towards low-power architectures such as the ones used in smartphone and tablets [12], we report in this paper the first results on the energy consumption of the DD-4DVAR algorithm on embedded processor. Note that our approach addresses the problem in the spirit of scalability analysis of parallel algorithms as distinct from practical performance analysis on specific architecture. We provide a mathematical analysis of the energy behaviour of the DD-4DVAR algorithm as function of the architectures characteristics of the platforms where are executed. The main objective is to capture the essential operations in the algorithm exhibiting a direct relationship with the measured energy. Such analysis will enable predicting the energy requirements of the DD-4DVAR code, provided that a set of architecture-dependent parameters are available, as well as understanding its energy breakdown, which may in turn underpin a systematic approach to combined performance/energy optimization. The experimental evaluation is carried out on a set of AMR based platforms made available by the Barcelona Supercomputing Center in the context of the Mont-Blanc European project [13]. The evaluation, aimed at understanding the energy breakdown and the related scalability issues, pointing out the importance of the underplay between parallel performance and energy optimization.

2 The DD-4DVAR Computational Kernel

Hereafter we provide a concise formalization of the DD-4DVAR model we implemented in Algoritm 1 [1].
Let t_k, $k = 0, 1, \ldots, n$ be a sequence of observation times and, for each k, let be

$$x_k \equiv x(t_k) \in \Re^N \tag{1}$$

the vector denoting the state of a sea system such that $x_k = \mathcal{M}_k\left(x_{k-1}\right)$ with $\mathcal{M}_k : \Re^N \mapsto \Re^N$ forecasting model.
At each time step t_k, let be

$$y_k = \mathcal{H}_k(x_k) \in \Re^p \tag{2}$$

the observations vector where $\mathcal{H}_k : \Re^N \mapsto \Re^p$ is a non-linear interpolation operator collecting the observations at time t_k.

The aim of DA problem is to find an optimal tradeoff between the current estimate of the system state (background) defined in (1) and the available observations y_k defined in (2).

Let (3) be an overlapping decomposition of the physical domain Ω such that $\Omega_i \cap \Omega_j = \Omega_{ij} \neq 0$ if Ω_i and Ω_j are adjacent and Ω_{ij} is called *o*verlapping region [1].

$$\Omega = \bigcup_{i=1}^{N_{sub}} \Omega_i \tag{3}$$

For a fixed time $t_k = t_0$, according to this decomposition, the DD-4DVAR computational model is a system of N_{sub} non-linear least square problems described in (4)-(5) where J_i in (5) is called cost-function.

$$x_0^{DA} = \sum_{i=1}^{N_{sub}} \tilde{x}_{0_i}^{DA}, \quad \text{with} \quad \tilde{x}_{0_i}^{DA} = \begin{cases} argmin_{x_0} J_i(x_{0_i}^{DA}) & on \quad \Omega_i \\ 0 & on \quad \Omega - \Omega_i \end{cases} \tag{4}$$

$$J_i(x_{0_i}^{DA}) = \|\mathbf{x}_{0_i}^{DA} - \mathbf{x}_{0_i}^{M}\|_{\mathbf{B_i}}^2 + \sum_{k=0}^{N} \|\mathcal{G}_{\mathbf{k_i}}(\mathbf{x}_{0_i}^{DA}) - \mathbf{y}_i\|_{\mathbf{R_i}}^2 + \|\mathbf{x}_{0_i}^{DA}/\Omega_{ij} - \mathbf{x}_{0_j}^{DA}/\Omega_{ij}\|_{\mathbf{B_{ij}}} \tag{5}$$

where $\mathcal{G}_k = \mathcal{M}_k \circ \mathcal{H}_k$.

x_0^{DA} in (4) is the *analysis* (i.e. the estimation of the vector $x_{0_i}^{DA}$ at time t_0). The variables $\mathbf{x}_{0_i}$ and $\mathbf{y}_{k_i}$ are the same vectors $\mathbf{x}_0$ and $\mathbf{y}_k$ in (1) and (2) defined on the subdomain Ω_i, $\mathbf{R_i}$ and $\mathbf{B_i}$ are the covariance matrices whose elements provide the estimate of the errors on y_{k_i} and on x_{0_i}, respectively.

Let $d = [y_k - \mathcal{H}(x_k)]$ be the *misfit*, by using the linearization of $\mathcal{H}$ such that $\mathcal{H}(x) = \mathcal{H}(x + \delta x) + H \ \delta x$, where H is the matrix obtained by the first order approximation of the Jacobian of $\mathcal{H}$ and, by setting $v_i = V_i^T \delta x_i$, with V_i such that $\mathbf{B_i} = V_i V_i^T$, the cost function in (5) is written as:

$$\begin{aligned} J_i(v_i) = \frac{1}{2} v_i^T v_i + \frac{1}{2} \sum_{k=0}^{N} (G_{k_i} V_i v_i - d_{k_i})^T R_{k_i}^{-1} (G_{k_i} V_i v_i - d_{k_i}) + \\ + \frac{1}{2} (V_{ij} v_i^{+} - V_{ij} v_i^{-})^T (V_{ij} v_i^{+} - V_{ij} v_i^{-}) \end{aligned} \tag{6}$$

The minimum of the cost function J_i in (6) is computed by the L-BFGS method [14] which implements a quasi Newton method. Then we need to compute $\nabla J_i(v_i)$ such that:

$$\nabla J_i(v_i) = v_i + \sum_{k=0}^{N} V_{k_i}^T G_{k_i}^T R_{k_i}^{-1} (G_{k_i} V_i v_i - d_{k_i}) \tag{7}$$

where $G_{k_i}^T$ is the adjoint operator of G_{k_i}.

Algorithm 1 The DD-4DVAR algorithm on each subdomain $\Omega_i \times [t_0, t_n]$

```
Input: $\{y_{k_i}\}_{k=0,\dots,m}$ and $x_{0_i}^M$
Define $H_{k_i}$
Compute $d_{k_i} \leftarrow y_{k_i} - H_{k_i} M_{k_i} \dots M_{1_i} x_{0_i}^M$    % compute the misfit
Define $R_{k_i}$ starting from the observed data $y_{k_i}$
Define $V_i$ starting from a temporal sequence of hystorical data $\{x_{k_i}^M\}_{k=0,\dots,M}$
Define the initial value of $\delta x_i^{DA}$
Compute $v_i \leftarrow V_i^T \delta x_i^{DA}$
repeat % start of the L-BFGS steps
    Send and Receive the boundary conditions from the adjacent domains
    Compute $J_i \leftarrow J_i(v_i)$    % Defined in (6)
    Compute $gradJ_i \leftarrow \nabla J_i(v_i)$    % Defined in (7)
    Compute new values for $v_i$
until (Convergence on $v_i$ is obtained) % end of the L-BFGS steps
Compute $x_i^{DA} \leftarrow x_{0_i}^M + V_i v_i$
```

3 Energy analysis of the algorithm

In this section we set a DD-4DVAR algorithm configuration and we perform a mathematical analysis of the energy behaviour of the algorithm.
For the DD-4DVAR algorithm configuration we assume:

- N defined in (1), which is the dimension of the problem, such that

$$N = n_x \times n_y \times n_z = n \times n \times 3$$

as this does not affect the generality, where $n \in \mathcal{N}$, $n > 1$;
- a 2D decomposition along the x-axes and the y-axes such that each subdomain has dimension:

$$N_i = \frac{n}{p} \times \frac{n}{p} \times 3; \qquad (8)$$

where $p \in \mathcal{N}$, $p > 1$. Then, N_{sub} the number of subdomain in (3) (which constitutes the domain decomposition) is

$$N_{sub} = p^2. \qquad (9)$$

- the algorithm be implemented on a parallel architecture by employing *nproc* processors such that $nproc = N_{sub}$, i.e. from (9), we are assuming

$$nproc = p^2.$$

Concerning the energy model, we assume that [8]:

- the energy consumption is *additive* and it is essentially proportional to the respective activity intensity in each component of the computing architecture, in terms of compute operation count, exchanged messages, memory accesses, *plus* a static energy contribution which is not affected by the activity and only depends on the considered time interval.

Based on the above assumption, we can write the energy breakdown as:

$$E^{HC}(p,n) = E_{comp}(p,n) + E_{mem}(p,n) + E_{mes}(p,n) + E_{static}(p,n) \tag{10}$$

where the superscript HC denotes the dependency on the computing architecture, and

- $E_{comp}(p,n)$ is the energy for computation:
$$E_{comp}(p,n) = E_d \cdot f^2 \cdot \mu_{comp}(p,n), \tag{11}$$
where E_d is a hardware constant [4], $\mu_{comp}(p,n)$ is the number of computations and f is the frequency;
- $E_{mem}(p,n)$ is the energy for memory accesses:
$$E_{mem}(p,n) = E_m \cdot \mu_{mem}(p,n), \tag{12}$$
where E_m is the energy consumed for a single memory access (both read and write) and and $\mu_{mem}(p,n)$ is the number of memory accesses;
- $E_{mes}(p,n)$ is the energy for message transfers:
$$E_{mes}(p,n) = E_t \cdot \mu_{mes}(p,n), \tag{13}$$
where E_t is the energy consumed for a single message transfer between the processors and $\mu_{mes}(p,n)$ is the number of message transfers at all processors;
- $E_{static}(p,n)$ is the static energy:
$$E_{static}(p,n) = E_l \cdot f \cdot T_{active}(p,n). \tag{14}$$
where E_l is a hardware constant [4] and $T_{active}(p,n)$ is the execution time for performing the whole algorithm.

Let

- $N_{L-BFGS,p}$ be the number of L-BFGS steps (see Steps 8 - 13 of Algorithm 1) which depends on the sub domains dimension (i.e., from (8), it depends on p) [2];
- n_C^{HC} be the maximum size of the allocable problem in the memory cache of the architecture HC.
- p_{max}^{HC} be the maximum number of cores of the architecture HC.

By assuming
$$n \leq n_C^{HC}, \quad p < p_{max}^{HC} \tag{15}$$
and by analyzing the time complexity of Algorithm 1, we can estimate the order of magnitude of the energy consumption by the following result.

Theorem 1. *By assuming (10), (11)-(14) and (15), it holds:*

$$E^{HC}(p,n) = \mathcal{O}\left(\mathcal{C}^{HC}(p) \cdot 9\frac{n^4}{p^2}\right) \tag{16}$$

where $E^{HC}(p,n)$ *denotes the energy consumption defined in (10) and where* $\mathcal{C}^{HC}(p)$*:*

$$\mathcal{C}^{HC}(p) = E_d \cdot N_{L-BFGS,p} + E_l \cdot t_{flop} \tag{17}$$

with t_{flop} *denotes the unitary time required for the execution in each processor of one floating point operation.*

Proof: *Let* $S_i(p,n)$ *and* $V_i(p,n)$ *denote the number of floating point exchanges at each algorithm iteration and the floating point computations at each iteration respectively, proportional to surface area and the volume of each subdomain in Algorithm 1:*

$$S_i(p,n) = 12\frac{n}{p} \tag{18}$$

$$V_i(p,n) = 3\frac{n^2}{p^2} \tag{19}$$

then $\mu_{comp}(p,n)$, $\mu_{mem}(n,p)$ *and* $\mu_{mes}(p,n)$ *are such that:*

$$\mu_{comp}(p,n) = N_{L-BFGS,p} \cdot p^2 \cdot V_i^2(p,n), \tag{20}$$

$$\mu_{mem}(p,n) = 2N_{L-BFGS,p} \cdot p^2 \cdot V_i(p,n), \tag{21}$$

$$\mu_{mes}(p,n) = N_{L-BFGS,p} \cdot p^2 \cdot S_i(p,n), \tag{22}$$

Also we assume $T_{active}(p,n)$ *be the execution time for performing* $V_i^2(p,n)$ *floating point operations:*

$$T_{active}(p,n) = t_{flop} \cdot V_i^2(p,n) \tag{23}$$

Then, from (10), (18)-(19) and (20)-(22), it holds

$$E^{HC}(p,n) = E_d \cdot N_{L-BFGS,p}\ (p^2)\left(3\frac{n^2}{p^2}\right)^2 \cdot f^2 + E_m \cdot 2N_{L-BFGS}\ (p^2)\left(3\frac{n^2}{p^2}\right) +$$

$$+E_t \cdot N_{L-BFGS,p}\ (p^2)\left(6\frac{n}{p} + 6\frac{n}{p}\right) + E_l \cdot t_{flop}(p^2)\left(3\frac{n^2}{p^2}\right)^2 \cdot f \tag{24}$$

As we run in a single computational node (i.e. $p < p_{max}$ *as expressed in (15)) this means that we are not implying communications, so the third term can be neglected. From qualitative observations, we can assume that the second term* *can be neglected because we fit the whole data in cache (as expressed in (15)), therefore a negligible number of access to the main memory are performed. Then the (16) follows.*

Definition 1 *(Energy Variation parameter) We denote with Energy Variation parameter the ratio*

$$VE_{p1,p2} = \frac{E^{HC}(p_1,n)}{E^{HC}(p_2,n)} \tag{25}$$

The following result holds:

Proposition 1 *For a fixed architecture and, under the hypothesis of Theorem 1, it is*

$$VE_{p1,p2} > \frac{p_2^2}{p_1^2} \tag{26}$$

for $p_2 \geq p_1$.

Proof: *From (24) and (16) for a fixed value of n, it is*

$$VE_{p1,p2} = \frac{\mathcal{C}^{HC}(p_1)}{\mathcal{C}^{HC}(p_2)} \frac{p_2^2}{p_1^2} \tag{27}$$

We observe that, from (27), it is

$$\frac{\mathcal{C}^{HC}(p_1)}{\mathcal{C}^{HC}(p_2)} > 1 \Longrightarrow VE_{p1,p2} \geq \frac{p_2^2}{p_1^2}$$

which gives:

$$\mathcal{C}^{HC}(p_1) > \mathcal{C}^{HC}(p_2) \Longrightarrow VE_{p1,p2} > \frac{p_2^2}{p_1^2} \tag{28}$$

From (28) and (17) it is

$$\mathcal{C}^{HC}(p_1) > \mathcal{C}^{HC}(p_2) \Longleftrightarrow E_d \cdot N_{L-BFGS,p_1} + E_l \cdot t_{flop} > E_d \cdot N_{L-BFGS,p_2} + E_l \cdot t_{flop}$$

As for a fixed architecture, the values of E_d, E_l and t_{flop} are also fixed, it is

$$\mathcal{C}^{HC}(p_1) > \mathcal{C}^{HC}(p_2) \Longleftrightarrow N_{L-BFGS,p_1} > N_{L-BFGS,p_2}$$

Due the better conditioning of the smaller problems, it is $N_{L-BFGS,p_1} > N_{L-BFGS,p_2}$ [2]. Then the (26) holds.

Remark 1 *We observe that, if the (15) is not satisfied, then $\mathcal{C}^{HC}(p)$ includes also E_{mes} which increases as the number of processors increases. In that case, for $p_2 > p_1$, it is:*

$$\mathcal{C}^{HC}(p_2) \geq \mathcal{C}^{HC}(p_1) \tag{29}$$

which gives

$$VE_{p1,p2} \leq \frac{p_2^2}{p_1^2} \tag{30}$$

4 Experimental results

The proposed approach is validated on a case study based on the linear Shallow Water Equation (SWE) for $n = 64$, i.e. we consider a fixed size configuration of the DD-4DVAR algorithm and we discuss results obtained by varying p.
The experiments are been conducted on architectures available at the Barcelona Supercomputing Center (BSC) and the power measurements have been enabled by the Mont-Blanc computing environment [13].

In table 1 are summarized the reference architectures. $HC = CT$ refers to a single Cavium ThunderX server [15], $HC = JT$ refers to a cluster of 16 nodes of Nvidia JetsonTx1, while $HC = MB$ refers to a partition of 5 nodes of the Mont-Blanc prototype cluster [13] used for this work.

	Cavium	**Nvidia**	**Samsung**
Specifications	**ThunderX**	**JetsonTx1**	**Exynos**
Instruction Set	ARMv8	ARMv8	ARMv7
Num. of cores / node	$2 \cdot 48$	4	2
Num. of cluster nodes	1	16	16
Clock freq. [GHz]	2.5	1.73	1.7
L2 cache size [MB]	16	2	1

Table 1. Reference architectures details

Relying on the potential of the Mont-Blanc computing environment, we were particularly interested in the results in terms of power efficiency and energy-to-solution. Here we provide results in terms of (measured) Energy Variation Parameter defined in (25) and computed using the values of energy consumptions given by $E^{HC}(p, 64) = P_p^{HC} \cdot T_p^{HC}$, where P_p^{HC} and T_p^{HC} are the power and the execution time respectively. We compare the obtained results with the upper and lower bounds provided in (26) and (30).
We observe that, in Table 1, the Cavium ThunderX has 16 Megabyte of memory cache which allows to satisfy condition in (15). In fact[3],

$$n_C^{CT} = 16 \cdot n_{C,1} = 96 > 64 = n, \quad p < p_{max}^{CT} = 2 \cdot 48 = 96.$$

Under condition (15), the (26) holds as confirmed by the results in Table 2.

[3] Due the time complexity of the computation, for each Megabyte, the values on n_C which is independent from the computing architecture, is such that: $n_{C,1} = \left\lceil \left(\frac{1048576}{8*3}\right)^{\frac{1}{6}} \right\rceil = 6$, where $\lceil \cdot \rceil$ denotes the integer part.

p^2	P_p^{CT}	T_p^{CT}	$E^{CT}(p, 64)$	$VE_{1,p}$	$p^2/1$
1	125.0 W	906 secs	113250.0 J	1.0	1
4	125.5 W	211 secs	26480.5 J	4.3	4
16	126.5 W	42 secs	5313.0 J	21.3	16

Table 2. Cavium ThunderX

The JetsonTx1 and Mont-Blanc, with 2 Megabyte and 1 Megabyte of cache instead (see Table 1) do not satisfy (15). In fact, $n_C^{JT} = 2 \cdot n_{C,1} = 12$ and $n_C^{MB} = 1 \cdot n_{C,1} = 6$ for the JT and MB respectively, both smaller than $n = 64$. In these cases, the upper bound in (30) holds as confirmed by the results in Table 3 and Table 4.

<table>
<tr><th colspan="6">$f = 800000$</th></tr>
<tr><th>p^2</th><th>P_p^{JT}</th><th>T_p^{JT}</th><th>$E^{JT}(p, 64)$</th><th>$VE_{1,p}$</th><th>$p^2/1$</th></tr>
<tr><td>1</td><td>5.3 W</td><td>429 secs</td><td>2273.7 J</td><td>1.0</td><td>1</td></tr>
<tr><td>4</td><td>6.6 W</td><td>115 secs</td><td>759.0 J</td><td>3.0</td><td>4</td></tr>
<tr><td>16</td><td>6.6 W</td><td>45 secs</td><td>297.0 J</td><td>7.7</td><td>16</td></tr>
<tr><th colspan="6">$f = 1700000$</th></tr>
<tr><th>p^2</th><th>P_p^{JT}</th><th>T_p^{JT}</th><th>$E^{JT}(p, 64)$</th><th>$VE_{1,p}$</th><th>$p^2/1$</th></tr>
<tr><td>1</td><td>6.5 W</td><td>210 secs</td><td>1365.4 J</td><td>1.0</td><td>1</td></tr>
<tr><td>4</td><td>10.0 W</td><td>86 secs</td><td>860.6 J</td><td>3.1</td><td>4</td></tr>
<tr><td>16</td><td>10.0 W</td><td>21 secs</td><td>210.0 J</td><td>6.5</td><td>16</td></tr>
</table>

Table 3. JetsonTx1

<table>
<tr><th colspan="6">$f = 800000$</th></tr>
<tr><th>p^2</th><th>P_p^{MB}</th><th>T_p^{MB}</th><th>$E^{MB}(p, 64)$</th><th>$VE_{1,p}$</th><th>$p^2/1$</th></tr>
<tr><td>1</td><td>5.4 W</td><td>375 secs</td><td>2025.0 J</td><td>1.0</td><td>1</td></tr>
<tr><td>4</td><td>5.5 W</td><td>86 secs</td><td>473.0 J</td><td>4.2</td><td>4</td></tr>
<tr><td>16</td><td>5.5 W</td><td>23 secs</td><td>126.5 J</td><td>16.0</td><td>16</td></tr>
<tr><th colspan="6">$f = 1700000$</th></tr>
<tr><th>p^2</th><th>P_p^{MB}</th><th>T_p^{MB}</th><th>$E^{MB}(p, 64)$</th><th>$VE_{1,p}$</th><th>$p^2/1$</th></tr>
<tr><td>1</td><td>5.4 W</td><td>181 secs</td><td>977.4 J</td><td>1.0</td><td>1</td></tr>
<tr><td>4</td><td>5.5 W</td><td>48 secs</td><td>264 J</td><td>3.7</td><td>4</td></tr>
<tr><td>16</td><td>5.5 W</td><td>13 secs</td><td>71.5 J</td><td>13.7</td><td>16</td></tr>
</table>

Table 4. Mont-Blanc

5 Conclusion

We introduced an energy analysis of the DD-4DVAR algorithm for data assimilation problems. An implementation of the algorithm was evaluated on some prototype ARM-based platforms made available by the Barcelona Supercomputing Center. We performed the analysis of the energy behaviour of the algorithm depending on several architectures characteristics. A preliminary experimental evaluation confirmed the estimations provided by our analysis on a fixed size problem varying the number of processors. As a future development, we aim at scaling up the methodology by demonstrating energy-driven parallelization approaches on production-grade ARM-based HPC clusters.

Acknowledgment

The research has received funding from European Commission under H2020-MSCA-RISE NASDAC project (grant agreement n. 691184) FP7 Mont-Blanc and Mont-Blanc 2 (grant agreements n. 288777 and 610402), H2020-FET Mont-Blanc 3 (grant agreement 671697).

References

1. R.Arcucci, L.D'Amore, L.Carracciuolo - On the problem-decomposition of scalable 4D-Var Data Assimilation models - Proc. of HPCS 2015, pp. 589-594, 2015.
2. R. Arcucci, L. D'Amore, J.Pistoia, R.Toumi, A.Murli - On the Variational Data Assimilation Problem Solving and Sensitivity Analysis, JCPH Vol. 335, pp. 311-326, 2017.
3. Carracciuolo L., D'Amore, L., Murli, A., Towards a parallel component for imaging in PETSc programming environment: A case study in 3-D echocardiography, Parallel Computing Vol 32, pp 67-83, 2006.
4. Anantha P. Chandrakasan, Samuel Sheng, and Robert W. Brodersen - Low-Power CMOS Digital Design - It Et JOURNAL OF SOLID-STATE CIRCUITS. Vol. 27, N. 4, 1992
5. L. D'Amore, R. Arcucci, L. Marcellino, A. Murli - HPC computation issues of the incremental 3D variational data assimilation scheme in OceanVar software. JNAIAM Vol 7(34), pp 91-105, 2012.
6. L. D'Amore, R. Arcucci, L. Carracciuolo, A. Murli - A scalable approach to variational data assimilation - J. Sci. Comput. 2, pp. 239-257, 2014
7. E. Kalnay - Atmospheric Modeling, Data Assimilation and Predictability. - Cambridge University Press, Cambridge, MA, 2003.
8. V. A. Korthikanti and G. Agha, Energy-Performance Trade-off Analysis of Parallel Algorithms, Hot Topics in Parallelism (HotPar), 2010.
9. G. Laccetti, M. Lapegna, V. Mele et al. - A Double Adaptive Algorithm for Multidimensional Integration on Multicore Based HPC Systems - Int. J. Parallel Programming 40, pp 397409, 2012.
10. A. Murli, V. Boccia, L. Carracciuolo et al. - Monitoring and migration of a PETSc-based parallel application for medical imaging in a grid computing PSE - Int. Federation for Inf. Proc., Vol. 39, pp. 421-432, 2007.

11. A. Murli, L. D'Amore, G. Laccetti et al. - A multi-grained distributed implementation of the parallel Block Conjugate Gradient algorithm - Concurrency and Computation: Practice and Experience Vol. 22, pp 2053-2072, 2010.
12. N. Rajovic, P. M. Carpenter, I. Gelado, N. Puzovic, A. Ramirez, and M. R. Valero, Supercomputing with commodity CPUs: Are mobile SoCs ready for HPC?, in High Performance Computing, Networking, Storage and Analysis (SC), 2013 International Conference for, 2013, pp. 1-12.
13. N. Rajovic et al., The Mont-Blanc Prototype: An Alternative Approach for HPC Systems, in Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis, Piscataway, NJ, USA, pp 38:138:12, 2016.
14. Nocedal, J., Byrd, R.H., Lu, P., Zhu, C.: L-BFGS-B: fortran subroutines for large-scale bound-constrained optimization. ACM Trans. Math. Softw. 23(4), 550560, 1997.
15. http://www.anandtech.com/show/10353/investigating-cavium-thunderx-48-arm-cores/