An efficient parallel implementation of cell mapping methods for MDOF systems

Nonlinear Dynamics manuscript No.
(will be inserted by the editor)

An efficient parallel implementation of Cell mapping methods for MDOF systems

Pierpaolo Belardinelli · **Stefano Lenci**

Received: date / Accepted: date

Abstract The long term behavior of dynamical system is usually analized by means of basins of attraction (BOA) and most often, in particular, with cell mapping methods that ensure a straightforward technique of approximation.

Unfortunately the construction of BOA requires large resources, especially for higher-dimensional systems, both in term of computational time and memory space.

In this paper, the implementation of cell mapping methods towards a distributed computing is undertaken; a new efficient parallel algorithm for the computation of large-scale basins of attraction is presented herein, also by addressing issues arising from the inner seriality related to the BOA construction. A cell-mapping core is thus wrapped in a management shell, in charge of the core administration, it permits to split over a multi-core environment the computing domain, by carrying out an efficient use of the distributed memory.

The proposed approach makes use of a double-step algorithm in order to generate, first, the multidimensional BOA of the system and, then to evaluate arbitrary 2D Poincaré sections of the hypercube that stores the information.

An analysis on a test system is performed by considering different dimensional grids; the effort of a parallel implementation towards medium and large clusters is balanced by a great results in terms of computational speed. The performances are strictly affected not only by the number of cores used to run the code, but in particular in the way they are instructed. To get the best from an implementation on a massive parallel architecture, the processes must be properly balanced between memory operations and numerical integrations.

P. Beladinelli
DICEA, Polytechnic University of Marche, Italy
E-mail: p.belardinelli@univpm.it

S. Lenci
DICEA, Polytechnic University of Marche, Italy
Tel.: +39-071-2204552
E-mail: lenci@univpm.it

A significant improvement in the elaboration time for a large computing domain is shown, a comparison with a serial code demonstrates the great potential of the application; the advantages given by the use of parallel reading/writing are also discussed with respect to the BOA grid dimension.

Keywords Parallel Programming · Cell Mapping Methods · Basins of attraction

1 Introduction

The global analysis of nonlinear dynamical system is a common methodology to investigate the response characteristics and complex phenomena [1]. As matter of that, the study of the attractors and the proprieties of the associated basins, such as their size and distribution, gives a useful overview in a multi-stability schemes [2].

It is not feasible to determine basins of attraction analytically, only some qualitative properties can be determined, thus the basins must be derived numerically.

When numerical techniques are practical applied in order to solve dynamical problems, the continuous state variables dealing with systems, have to be regarded as discrete quantities. The instrumentation of a code, thus, introduces discretization errors as well as roundoff errors due to hardware limits. The implementation of some of the existing routines to determine the basins of attraction are not free of these problems, moreover they present limitations to utilize the parallel processing capabilities of modern architectures. Given a system modelled by a set of ODE's, the determination of stable (attractors) or unstable (saddle solutions and repellors) solutions with their owns domains (or basins) of attraction (see e.g.[3]), becomes computational effortly in high-dimensional systems.

Due to the large computational costs, several attempts have been done to improve the elaboration techniques for building basins of attraction. Numerical tools have been proposed, see e.g. [4], but at the best of our knowledge they are not suited to address the request of efficiency for large scale system. Furthermore their implementation in multicore environment or HPC systems are rather involved; this challenge is undertaken in this manuscript by presenting an algorithm in order to implement a cell mapping core on a massive multicore structure.

2 Literature survey

A common used method undertaken in the determination of the global overview of attractors and basins of attraction is the so-called grid of starts (GOS) or grid of points (IGP) [5]. The state space of the initial conditions is divided into a discrete set of cells, indivisible entities of the state of the system. All the possible states within the cell refer to the same condition, commonly addressed in the center of the cell. Starting from each initial point in the grid, the basin is derived identifying the attractor of each starting point. The full long-time numerical integration up to the steady-state behaviour of all the trajectories, is large computational costing and thus, although the algorithm, with sufficient transient time, gives a very good accuracy [6], it is not suitable, in its original form, for large or high-dimensional problems [7].

A valid alternative to GOS routines is represented by the cell mapping methods [1], that have been introduced by Hsu in the '80s [8,9]. The general idea of cell mappings is to define the mapping between cells by means of numerical integrations over a small time step in order to approximate all the possible trajectories. Over the years, a variety of different cell mapping methods have been developed. The first, the Hsu simple cell mapping (SCM), is based on the center point method [10]: first, as in the IGP, the state space is discretized and the center point of each cell is calculated. The image cell which contains the end point of the calculated trajectory, after an integration over a period Δt, is then detrminated. Only for autonomous system the choice of the period is arbirary, while, for periodic cases, the interval between the cell inspection must be equal to the periodicity of the system. Although with SCM the computing time with respect to the GOS is strongly reduced, if the cell size is not sufficiently small, a propagation of error can occur. The center point method creates an inconsistency between the endpoint of a trajectory segment and the start of the subsequent trajectory. Furthermore, the presence of fractalities along the domain boundaries and the possibility of chaotic behavior represent weak points of SCM. In order to overcome these limitations the generalized cell mapping (GCM) has been developed by Hsu et al. [11,12]. Differently from the SCM, each cell is mapped onto more than one cell, the method provides also a probabilistic analysis of the system suited for the description of chaotic behaviour with a slight increment of the computational cost with respect to SCM. An algorithm for the approximation of the GCM, with a deterministic a priori error estimation, is presented in [13] by Guder and coworkers. Engineering applications of CM methods are numerous: with SCM, Sun [14], Crespo and Sun [15], studied the optimal control of deterministic systems, while the multiobjective optimal design of full state feedback controls is presented by Xiong et al. [16] by using the simple cell mapping method with an hybrid algorithm. The research developments of cell mapping method for systems control are summarized in [17], presenting several main cell mapping methods. The study of fuzzy chaotic attractors and their basins of attraction are undertaken by Hong and Sun with GCM [18,19]

The introduction of the interpolated cell mapping (ICM) was especially to solve the spurius results in fractal basins. The ICM, developed by Tongue [5,20] and Tongue and Gu [21], is quite different from SCM and GCM, by consisting in interpolation steps along with integrations that concern the grid points surrounding the endpoint of an integrated trajectory segment. The accuracy is increased by the use of interpolation steps that, however, for high-dimensional systems, adds CPU time slowing the routine [6]. In Ge and Lee [22] the ICM is used to analyze random dynamical systems.

Recently, several modification on ICM have been presented, such as the higher-order method of ICM, namely the tensor product interpolated cell mapping (TPICM) that uses a modified more complex mapping [23]. Another modification is represented by the multiple mapping (MM) [20]. With the MM the regular mapping over one period is subdivided by two maps corresponding to half a period each. The state space distorsion is thus limited and the chaotic attractor is well determined; these facts have increased the practical importance of the method along with only a slight gain on the computational time.

A combination of MM with the ICM, the mixed cell mapping (MCM), was predented by van der Spek [24] . The MCM is based on the ICM but use the MM in some situations such as the divergence of the interpolated trajectories.

A serious issues to be address, due to the special attention on the computational efficiency, is the parametric study on basins of attraction. Continuation methods are commonly used to study the influence of parameters in dynamical systems. The analogous methods that elaborate the evolution of the domains boundaries when a system parameter is varied, are the parameter variation (PV) methods for cell mapping. The two most important are the extensions of the SCM and ICM, namely the PVSCM and the PVICM respectively [25,26]. Thanks to the PV methods the time to perform a parametric analysis is less than that occurred to construct a new basins by considering the updated parameter.

All the methods up to here presented are not adeguate for large problem or system with many degrees-of-freedom. To overcome this limitation, van Campen, van der Spek and coauthors [27,28] developed the multi-degrees-of-freedom cell mapping method (MDCM). The method has been succesfully applied by Crespo et al. in [29] and in [30] by Van Campen and coworkers. The aim of the MDCM method is to determine the intersections of the basins of attraction of a N-dimensional space with a two-dimensional subspace. The MDCM can menage systems with an arbitrarily large number of dimensions by using a coordinate numbering convention [26]. The technique processes only the cells in the two-dimensional subspace corresponding to the basin portrait and the cells encountered in each subsequent trajectory leading to the attractor. Since the MDCM uses computational resources only for the two-dimensional portraits, it scales efficiently to high-dimensional systems. Cell mapping for multi-degree-of-freedom-systems has been implemented by Kreuzer and Lagemann [31] in parallel computers with an adaptive cell refinement able to reduce the total number of cells.

The inner seriality of the method, and the consequent inefficiency in modern multicores computers, has been overcome by Eason et al. [32] with a modified version of the MDCM method called parallelized multi-degrees-of-freedom cell mapping (PMDCM) method. The mofication of the MDCM algorithm permits the use of parallel computing resources [33].

In spite of the last aforementioned methodologies, applications in high-dimension systems are still rather scarce. A four-dimensional phase space is investigated in [34] by presenting two-dimensional cross-sections basins for the oscillation and rotation of a double pendulum. Sections of six-dimensional basins of attraction, describing the nonlinear dynamic behaviour of the clamped-free beam subjected to a harmonic axial load, are carried out by Carvalho et al. in [35]. The evolution of the basins of attraction in a four-dimension system, is used in the work of Goncalves et al. [36] to perform a global stability analysis of a parametrically excited cylindrical shell.

3 Algorithm description

Given an n-dimensional system of ODEs, the elaboration of multidimensional basins of attraction is usually tackled by the investigation of low-dimension sections, i.e.

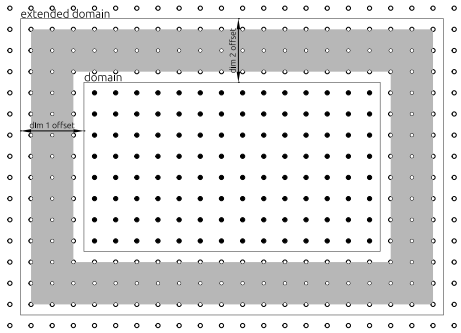


Fig. 1: Computing domain with its extension (2D)

the basin is not fully determinated in all the system dimensions. If the representation concerned by means of 2D Poincaré sections guarantees a quite reliable representation, it cannot be a deterministic tool for the dynamical analysis of the entire system, in example, integrity measures have to work in the whole hyper-space of the basin.

As a consequence of that first a binary file is created with all the informations about the basins,and only in a second time, proper sections of bidimensional basins are performed. This approach, even if might looks quite intricate, permits multiple accesses to a computed multidimensional basin and to store efficiently the data. This approach can be also very useful for PV analysis, i.e. to analyze the evolution of the boundaries in multiple dimensions. Let $\dot{\mathbf{y}} = \mathbf{f}(t, \mathbf{y})$ be a system of first-order ordinary differential equations in time t with $\mathbf{f} : \Psi \subset \mathbb{R} \times \mathbb{R}^n \to \mathbb{R}^n$, where n is the spatial dimension of the problem and dot stands for the time derivative. Given proper boundaries in each dimension, a partitioning of the state space leads to the discretized cell state space. Follows that the ensemble of initial conditions to be investigated grows exponentially witn n and it is a function of the discretization density in all the dimensions. Furthermore since during the integration the path of each orbit must be followed to verify a trajectory divergence [4], each dimension limit is also extended accordingly (see Fig. 1) making harder the total domain limitation.

The presented approach is aimed to reduce at minimum the waste of memory in the code parallelization; the problem is splitted onto several nodes by means of a distributed memory syntax where the MPI programming interface [37,38] is used. The subset for each memory process is a part of the whole computing domain (see Fig. 2(a)). By means of an unrolling procedure, shown in Fig. 2(b) each cell can be identified by an unique index.

The code scheme performing the multidimensional basin is sketched in Fig. 3 An initial routine has to set-up properly the resources; the software is explicitly aimed to be able to exploit the hardware characteristics: the program controls if the MPI settings join the resources requested and, in case of control failure the code stops. Also the data setting must be checked, in particular, the coherence of the boundaries settings, the system dimension and the number of the cells. An automatic reallocation routine tries to fix errors in order to go ahead towards the computing algorithm. The

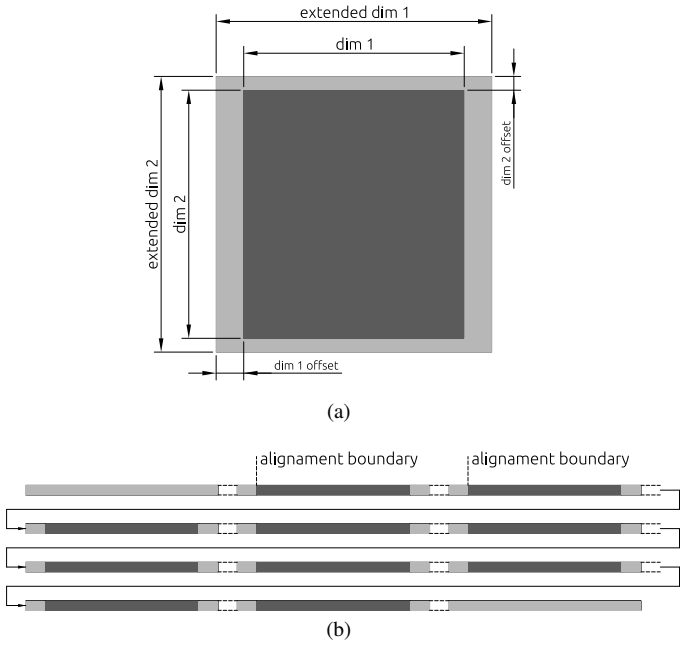


Fig. 2: The data structure (a) and its unrolling (b)

program has a standard flux that terminates with the kill of all processes and with the finalization for the MPI functions as soon as the file containing the basin is generated.

The computing algorithm, herein summarized in Fig. 4, consists in a connection network between three main characters. A master process supervises with synchronization e coherence operations the computation; it collects the status of each process and starts or stops actions on each core.The mere computation is performed by the integrator processes, they read and store the information by exploiting memory cores. This architecture, shown in Fig. 4 permits, by reallocating the data, to change the balance between the elaboration cores and memory managers in order to take advantage of the hardware. In details, the distribution of the memory processes and integrators must be balanced in order to obtain good performances as shown in Sec. 4. Exchanging data between processes, especially if belong to different nodes, represents a key point in the success of the parallelism.

The Cell Mapping (CM) core, that is the real computing routine, is within the integrator processes (Fig. 5). Up to the master declares the process valid, the integrator calls the actual status of the specific cell to be elaborated. It can lie on a whatever slice of memory and it could be interrogated by any other integrator at the same time: it follows that a fragmentation on the data set of ics helps to spread out the MPI calls. Otherwise a full compact slice of memory is preferred to have an easy update and coherence. The cell status regulates the action to be performed, also with respect to the CM parameters and the method implemented as described in Sec. 2. For further

details in the code listing for a simple CM method is suggested to refer to [4]. Since the CM method does not affect the cell management, it is modifiable without any influence on the shell management that administrates the exchange of the data between the processes.

In the memory processes are initially allocated the memory areas needed to store the cells informations. The master subdivides the total amount of cells with respect to the number of processes that will be dedicated to memory operations. Each process collects the integrators request by means of non blocking receive functions, it is in charge to update a cell after an integration step, and also to both read and inform the asking process with a cell status (see Fig. 6). After the discovery of an attractor, the master informs all the processes that have to update their private database. As described by Belardinelli et al. in [7], with a distribuited computing approach, a coherence between the nodes is mandatory to share the informations about the attractors. Two schemes of implementation can be applied, based on a real-time synchronization between the computing nodes or with *a posteriori* processing. If a coherence stencil is applied at the end of the process, the information about the recursion of the trajectories must be stored and consequently elaborated. A global operation, is carried out sharing the attractors, by associating their basins, and processing the recursion of the trajectories. The effect of *a posteriori* processing is shown in Fig. 7. It has to be remarked the synchronization is performed by all the memory processes in parallel, i.e. almost simultaneously, and only on the respective slice of memory. By means of a real time processing, the attractors information are shared immediately, the final coherence is thus no longer needed. With respect to a GOS alghoritm, the times to perform the two aforementioned procedures are rather similar, since the integration times are much more less of IGP, and the delay caused in the integration interruption by the message passing is damped.

A further task is the dimension of the packets of ics sent by the masters towards the integrators. It has been proof by several runs that the best choice is to organize the amount of ics proportional with respect to the total grid dimension. The equal repartition between the cores is not suggestible in order to obtain the best performance in term of computational time spent. Big data packets leads to less request of ics from the integration cores, but the risk of an asynchronous finalization occur, otherwise an high frequency in the communication with the master represent a bottleneck in the flux of the code.

Finally, when all the initial conditions has been analyzed, the master runs the routine for the binary file generation. It makes optimum use of the MPI I/O writing in order to reduces at maximum the time wasting for an external writing, usually a very slow operation. The memory cores access in a portion of the whole file; each memory area is unphased between different processes and thus the writing is greatly faster with respect a standard writing.

It has to be remarked that the writing operation is not a mandatory condition for the whole basin. If a proprietary cluster is used the data, that for a multidimensional basis is usually very large, can be held splitted on different nodes: this allows to store a big amount of data. The cross section generator can access, in this case, on the spread data and generate then a 2D figure.

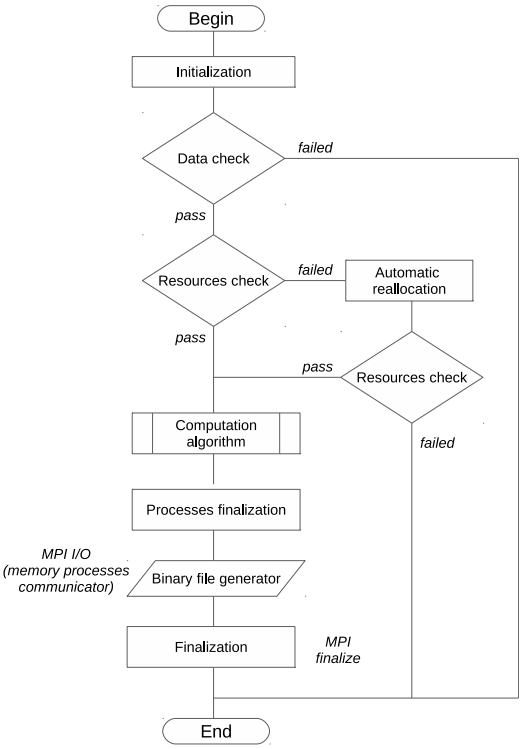


Fig. 3: Flowchart with the steps towards the computing algorithm

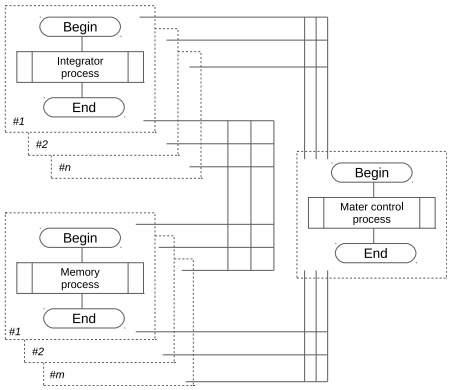


Fig. 4: The computing algorithm

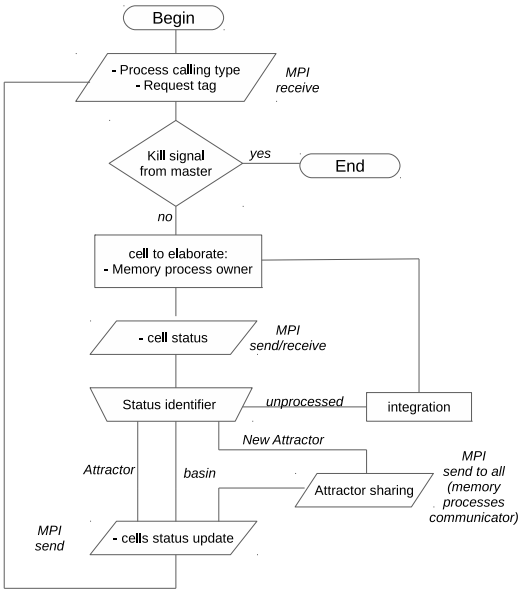


Fig. 5: The integrator process

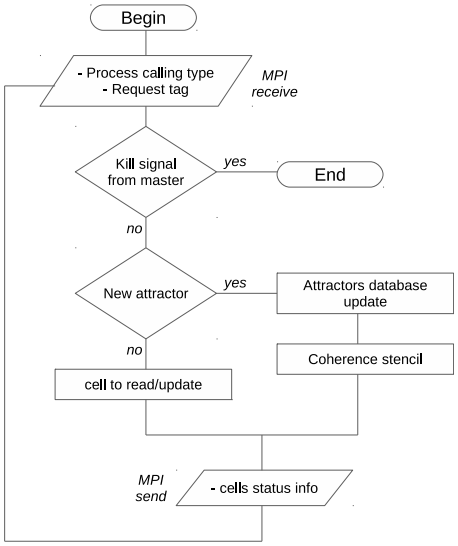
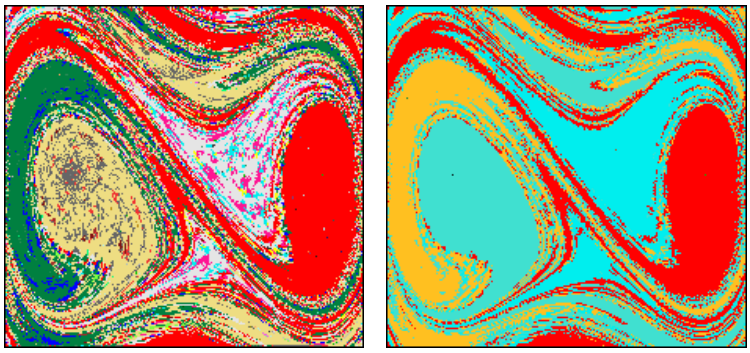


Fig. 6: The memory process



(a) Before (b) After

Fig. 7: The coherence stencil

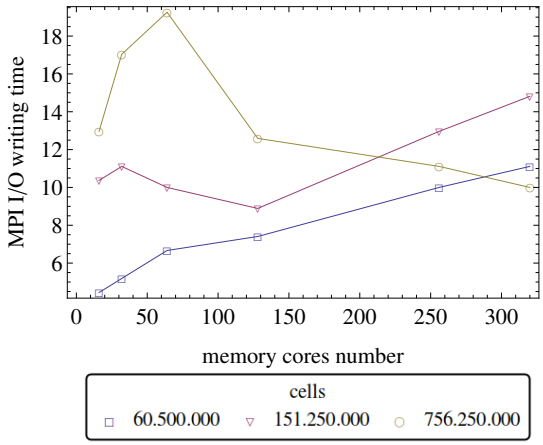


Fig. 8: The parallel writing of the binary file

In Fig. 8 is reported the time spent to generate the binary file with respec to the number of memory processes for different grid dimension.

As shown the MPI I/O helps to decrease and optimize the time especially if large grid are used. The serial writing is preferred for reduced grids.

Up to here the basins data are generated and can be used for different purposes, as said for example PV or dynamical integrity analysis. In order to have qualitative view the data must be interrogated with the aim to create an intelligible image. The MPI settings for the cross-section generator are completely independent from the binary generator. This allows to run the software in a different machine, commonly less powerful, or to schedule multiple sections at the same time. Obviously if the sections are created independently, the data should check the created basin, such as the system dimension and boundary limits; it must be moreover indicated the number of cores involved,the data paths the indication of the section to plot.

The access to binary file happens simultaneously by all the process in order to reduce the loading time, as said it has to be noted that the file contains the basins for all the dimensions thus can be very large. The parallel reading permits also to split the data across multiple cores: it is very well posed for shared-memory platform.

As soon as all the data are loaded, a routine realizes in each core the list of the addresses of the cells to be plot. It runs a multidimensional slice, on the hypercube given in the binary file, with the plane of the cross-section selected.

It has been verified by the authors that in the writing of the cross-section, the parallel writing is not necessary up to very large dimensions for the 2D cross-section. The implementation of a parallel writing activation if, and only if, a dimension threshold is overcome will be implemented as a further development.

4 Results

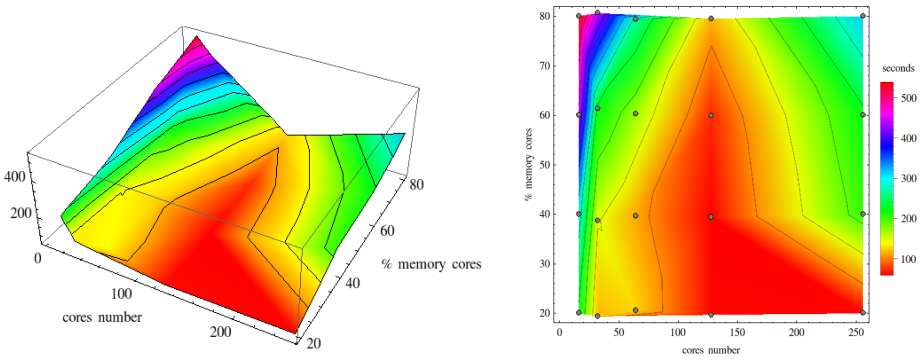
In this part of the paper the results of a parametric analysis testing the routine are reported. As above described, the code permits to set the best balance between the number of cores in charge of the memory: the efficiency of the of the elaboration changes accordingly. In Figs. 9-11 we report density diagrams with the computational time spent to generate the binary file with the basin data. The figures differ the one from the other for the grid dimension; they show the computational cost as function of the total number of cores with respect to the percentage of memory cores used. Figures from 9 to 11 highlight several areas of variation of the computational times. These behaviors depend on the saturation of the message passing that is a bottleneck for the time integration. If a perfect balance between memory cores and integrators is found, the code runs efficiently and a minimum in the elaboration time occurs. For a massive number of cells the use of a grater percentage of memory cores permits to split better the multiple integrations across a large number of cores.

A comparison of the presented code with the software [4] is reported in Table 1. The code, being pure serial, scales almost linearly with the CPU frequency clock. By using the parallel approach[1] the computational time is greater reduced. Furthermore it is wort to observe that [4] performs a 2D Poincaré section of a system, but it does not elaborate completely a multidimensional basin. The same effort can be exploited by the parallel code to describe a larger 2D section, or to perform an analysis of a multidimensional system; subsequently, an arbitrary number of sections can be easily carried out by means of the cross-section generator.

5 Conclusions

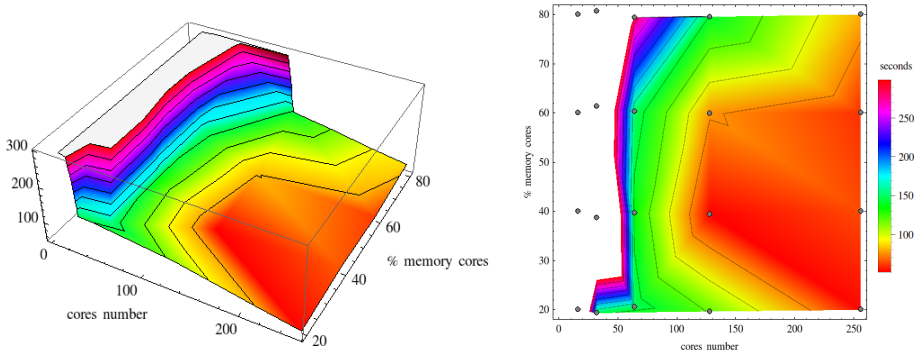
The effort and the complexity to implement an efficient cell-mapping method in a parallel platform cannot be considered negligible: the computation must be instructed properly in order to obtain the best performances. To undertake high-dimensional problem a parallel implementation in a multicores environment is resulted to be the

[1] Cluster EURORA@CINECA: 64 compute nodes (32 nodes with a 2 eight-core Intel(R) Xeon(R) CPU E5-2658 @ 2.10GHz and 32 nodes with 2 eight-core Intel(R) Xeon(R) CPU E5-2687W @ 3.10GHz)



(a) (b)

Fig. 9: Computational times with respect to the number of both memory and total cores. System dimension: 60500000 cells



(a) (b)

Fig. 10: Computational times with respect to the number of both memory and total cores. System dimension: 151250000 cells

Code	Present	Ref. [4]	Ref. [4]
System	EURORA[1] 128 cores 50 mem cores	Intel(R) Celeron(R) B830 CPU @ 1.80GHz	Intel(R) Core(TM) i5-2410M CPU @ 2.30GHz
Time [s]	39	256	159

Table 1: Computational times by using parallel and serial approach. Domain grid dimension 10^6 cells.

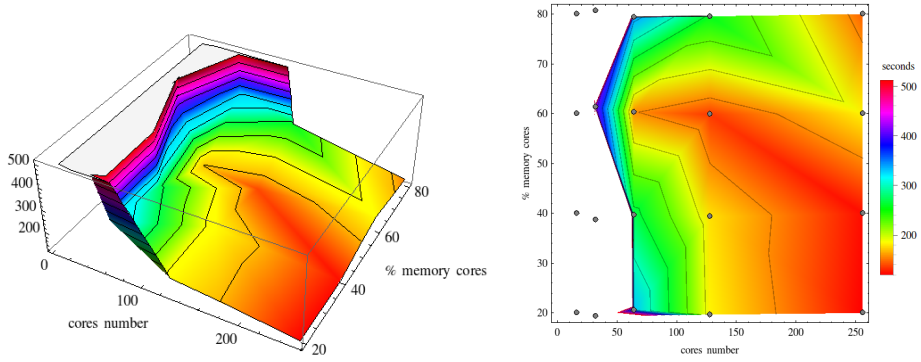


Fig. 11: Computational times with respect to the number of both memory and total cores. System dimension: 756250000 cells

best choice: it has been demonstrated by very low computational times with respect to the serial approach.

The proposed manner to structure the communication between a large number of cores uses a specific purpose identification such as integrators and memory processes.

It has been found that the optimal time has a double dependance, on the system-dimension and on the subdivision of cores in charge of the integration or memory operations.

Future works will involve an hybrid algorithm able to exploit several layers of parallelism today available in a modern hybrid architectures and supercomputers.

References

1. J. Sun, A. Luo, *Global Analysis of Nonlinear Dynamics*. Nonlinear Systems and Complexity (Springer, 2012)
2. G. Rega, S. Lenci, Nonlinear Analysis: Theory, Methods & Applications **63**(5-7), 902 (2005)
3. J. Thompson, H. Stewart, *Nonlinear Dynamics and Chaos* (Wiley, 2002)
4. H. Nusse, J. Yorke, *Dynamics: Numerical Explorations* (Springer, 1998)
5. B. Tongue, Physica D: Nonlinear Phenomena **28**(3), 401 (1987). DOI http://dx.doi.org/10.1016/0167-2789(87)90028-5. URL http://www.sciencedirect.com/science/article/pii/0167278987900285
6. Z.M. Ge, S.C. Lee, Journal of Sound and Vibration **199**(2), 189 (1997)
7. P. Belardinelli, S. Lenci, International Journal of Non-Linear Mechanics **submitted** (2015)
8. C. Hsu, Journal of Applied Mechanics **47**(4), 931 (1980)
9. C. Hsu, R. Guttalu, Journal of Applied Mechanics, Transactions ASME **47**(4), 940 (1980)
10. C. Hsu, *Cell to Cell Mapping; A Method of Global Analysis for Nonlinear System* (Springer-Verlag, 1987)
11. C. Hsu, Journal of Applied Mechanics, Transactions ASME **49**(4), 895 (1982)
12. C. Hsu, R. Guttalu, W. Zhu, Journal of Applied Mechanics, Transactions ASME **49**(4), 885 (1982)
13. R. Guder, M. Dellnitz, E. Kreuzer, Chaos, Solitons & Fractals **8**(4), 525 (1997)
14. J. Sun, Advances in Intelligent Systems and Computing **175 ADVANCES**, 3 (2013)
15. L. Crespo, J. Sun, Nonlinear Dynamics **31**(2), 119 (2003)
16. F.R. Xiong, Z.C. Qin, Y. Xue, O. Schtze, Q. Ding, J. Sun, Communications in Nonlinear Science and Numerical Simulation **19**(5), 1465 (2014)
17. W. Xu, C. Sun, J. Sun, Q. He, Advances in Mechanics **43**(1), 91 (2013)

18. L. Hong, J. Sun, International Journal of Bifurcation and Chaos **16**(10), 3043 (2006)
19. L. Hong, J. Sun, Chaos, Solitons and Fractals **27**(4), 895 (2006)
20. B. Tongue, International Journal of Non-Linear Mechanics **25**(2-3), 177 (1990)
21. B. Tongue, K. Gu, Journal of Applied Mechanics, Transactions ASME **55**(2), 461 (1988)
22. Z.M. Ge, S.C. Lee, Journal of Sound and Vibration **194**(4), 521 (1996)
23. B. Tongue, K. Gu, Journal of Sound and Vibration **125**(1), 169 (1988)
24. J. van der Spek, Cell mapping methods: modification and extensions. Ph.D. thesis, Technical University of Eindhoven (1994)
25. J. van der Spek, C. de Hoon, A. de Kraker, D. van Campen, Nonlinear Dynamics **7**(3), 273 (1995)
26. M. Wiercigroch, B. de Kraker, *Applied Nonlinear Dynamics and Chaos of Mechanical Systems with Discontinuities*. Series in Nonlinear Science, Series A, Volume 28 (World Scientific, 2000)
27. D. Van Campen, E. Van De Vorst, J. van Der Spek, A. De Kraker, Nonlinear Dynamics **8**(4), 453 (1995)
28. J. van der Spek, D. van Campen, A. de Kraker, in *Proceedings of the 1994 International Mechanical Engineering Congress and Exposition*, vol. 192 (Chicago, IL, 1994), vol. 192, pp. 151–159
29. L. Crespo, J. Sun, Nonlinear Dynamics **28**(3-4), 323 (2002)
30. D. Van Campen, A. De Kraker, R. Fey, E. Van De Vorst, J. Van Der Spek, Chaos, Solitons and Fractals **8**(4 SPEC. ISS.), 455 (1997)
31. E. Kreuzer, B. Lagemann, Chaos, Solitons & Fractals **7**(10), 1683 (1996)
32. R. Eason, A. Dick, Nonlinear Dynamics pp. 1–13 (2014). DOI 10.1007/s11071-014-1310-8
33. R. Eason, A. Dick, S. Nagarajaiah, Journal of Sound and Vibration **333**(15), 3490 (2014)
34. M. Marszal, K. Jankowski, P. Perlikowski, T. Kapitaniak, Mathematical Problems in Engineering **2014** (2014)
35. E. Carvalho, P. Goncalves, G. Rega, Z. Del Prado, Shock and Vibration **20**, 1073 (2013)
36. P. Gonalves, F. Silva, Z. Del Prado, Nonlinear Dynamics **50**(1-2), 121 (2007)
37. M.P.I. Forum, *MPI: A Message-Passing Interface Standard, Version 3.0* (High Performance Computing Center Stuttgart, 2012)
38. M. Snir, S. Otto, S. Huss-Lederman, D. Walker, J. Dongarra, *MPI: The Complete Reference* (MIT Press, 1996)