An Analytical Data Management as a Cloud Service for Numerical Simulations

Ramon G. Costa, Fábio Porto, Bruno Schulze

[1]{ramongc, fporto, schulze}@lncc.br
Laboratório Nacional de Computação Científica - LNCC - Petrópolis/RJ, Brazil

Abstract. *Numerical simulation of natural phenomena is being fostered by recent advances in powerful high processing computing platforms. Scientists in various areas, such as human cardiovascular system, model a phenomenon being studied through a set of mathematical equations. The latter are transformed into a computing model, using one of the available numerical methods. As scientists strive to obtain a more realistic simulation, a huge amount of data is produced. Unfortunately, there has been little work on supporting numerical simulation data management, which leaves simulation scientists with huge standard text files and complex analytical programs that eventually extract some meaningful information to validate scientific hypotheses. In this context, this paper tries to bridge this gap by raising some issues involved in numerical simulation data analysis. A representation for numerical simulation data is presented that considers a multidimensional model, for dimensional variables, and their corresponding physical quantities. An initial set of analytical operators are identified and their semantics discussed. The SciDB system is used to implement a first prototype supporting the human cardiovascular system simulation developed at the LNCC. Additionally, a cloud service to interface with the numerical simulation data manager is proposed and its integration with the Neblina cloud middleware is explored. We expect that this work will provide a better understanding concerning the needs involved in analytical data management for multidimensional numerical simulations.*

1. Introduction

Many scientific areas are taking advantage of development in high processing computing to model natural phenomena through in-silico simulation. The analysis and observation of computer simulation results allow scientists to validate their scientific hypotheses [10] enhancing their understating about the phenomenon.

The process involved in creating a scientific simulation is complex. It starts by observing phenomenon data and modeling the process through a set of mathematical differential equations that expresses the variation of selected physical quantities on time-space. Next, the scientist may choose an appropriate numerical method that would solve the equations and compute for each reference point the values for selected physical quantities. In simulations of natural phenomena the more fine grained the time interval and smaller the space partitions are, the more continuous, therein realistic, the simulation would be. Using state-of-art cluster platforms computer scientists strive to obtain the most possible realistic simulations, to study the human physiological system, in medical applications, or to compute weather forecasts, just to name a few applications. From a data management viewpoint, a huge amount of data is being produced by computer simulations and as more accurate become the simulations the worst performance is going to be.

Solvers consume and produce huge text files whose data must be analyzed through specialized programs. Validating scientific hypothesis becomes a huge challenge to which dedicated applications must be built to answer each new specific question scientist may raise. Furthermore, the lack of a guiding data model leave non database specialists with few resources either than modeling data in ad-hoc fashion, making sharing of results and their usage a challenge. Finally, as the processing capacity of clouds, clusters and multi-core machines increases share-nothing parallelism programming models offer great potential to reduce the elapsed-time of simulation results analysis. In cases in which domain partition is possible, internal support for data partitioning and replication would be essential.

In this paper we propose a novel strategy for scientific simulation data management based on database approach. We focus on the support to the analysis of simulation results, taking a sample of a simulation output and loading it onto a cloud data service modeled to manage space-time multi-scale dependent data. The data is modeled using a multi-dimensional array representation for space and time and new operations are identified to support simulation data analysis. As a first class of operations, we are interested in those expressing causal relationships between simulation states, as proposed by Sowa [13] and discussed in the context of data provenance by Meliou et al. in [1]. We illustrate our discussion with the simulation of the human cardiovascular system developed at LNCC, INCT-MACC [11].

2. A simulation of the human cardiovascular system

The numerical simulation is the type of simulation that uses numerical methods to quantitatively represent the evolution of a physical system and from the simulation results to draw appropriate conclusions, obtaining a better understanding of the system.

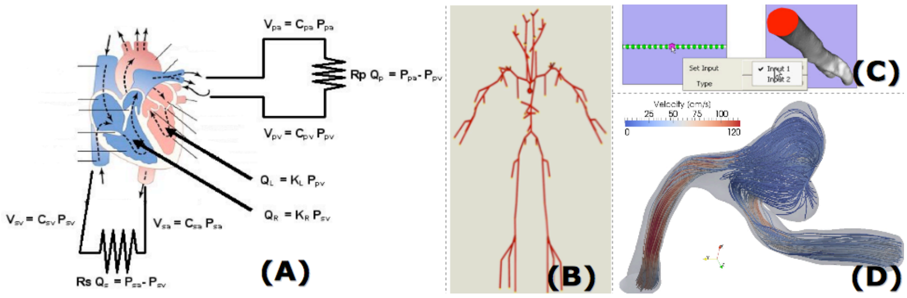


Figure 1. Models: (a) 0D; (b) 1D; (c) coupled; and (d) 3D

As an example, the 0D, 1D and 3D models used in the HeMoLab [6], can be seen as different scales, with different representation schemas. In a 0D model, a single point value is compute for pressure, flow and volume. Fig. 1a shows both heart chambers, defined by a system of seven equations. The 1D model, Fig. 1b, represents the human cardiovascular system using line segments. The data output of the 1D model is used as input for the 3D model, producing a coupled model (Fig. 1c). Finally, in a 3D model computation is based on a mesh structure, Fig. 1d.

3. Literature review

The investigation on data management techniques in supporting to scientific applications is not new. Even so, recent advances on instrumentation and high processing computing infrastructure has called for a revision of old techniques and the investigation of new ones.

In this section, we discuss some of this recent work involved in managing and processing scientific data and, more specifically, data produced out of numerical simulation.

Recent studies have demonstrated the need for a more efficient storage, indexing and processing strategies for scientific data [2] e [9]. Stonebraker [5] argues that the current data management technology is clearly unable to solve the demands of scientists. Scientific experiments and analysis are executed using scientific workflows, which focus their design on the activities and their execution order, leaving data modeling to be done on a design-by-design basis. In Ogasawara [2], scientific workflows are modeled as data intensive applications. Parameter sweep experiments evaluate data represented as a set of parameter configuration values. Moreover, typical workflow activities are identified according to their data consumption and production rate and mapped to algebraic operators, such as: Map, Reduce, Filter, and JoinQuery.

The storage of numerical simulation data has been investigated by [9]. In their work, multidimensional scientific data are modeled using an array data model. Thus, storing and processing large matrices become the main issue. The authors propose different types of array storage models according to array sparsity. The authors discuss the use of the Compressed Row Store and the B-Tree structures adapted to support the array data model and to save dead spaces. Moreover, the authors discuss strategies to efficiently support some operations over multi-dimensional arrays. The support for matrix multiplication may benefit from data allocation that may appear structured in rows and columns, while the Fourier Fast Transform (FFT) operations are backed up by a matrix linearization method based on bit-reversal order technique [9]. In our work, our focus is rather on an adequate representation of simulated data in support to simulation results analysis.

An important initiative in support for scientific data management is SciDB [7], a database management system for scientific applications. It has been designed as a collaboration among scientific researchers, mostly astronomers, and database scientists [5]. It offers a multidimensional model based on multiarray representation. Its functionality includes: data versioning, uniform distribution of data across the nodes of a cluster, and two query language interfaces: AFL and AQL languages [12].

Considering the analysis of simulation data, an interesting question is to determine the past and future cone of information, as presented by Sowa [13]. Indeed, depending on the mathematical model adopted, one may be interested in explaining the path of a given particle traveling on a physical domain. This is also considered a kind of retrospective provenance information [4] type of analysis. Similar to this is the study of causality in databases [1], in which data that contributes to a given result is considered to cause, with a certain responsibility, such result. In the context of numerical simulation, the identification of points in the physical space and time that were used in computing other points later in time are considered to be their cause. This may be used to identify the reason for some anomalous behavior on the simulated phenomenon.

From an architectural viewpoint, scientific projects pose some important requirements. Firstly, scientific cooperations require data to be available to and from different localities. In particular, scientific workflows should run at a locality close to its data. In case of numerical simulation, the locality where data is produced may dictate the execution allocation. As the accuracy of simulations increases, more processing will be required. In this context, granting extensions of processing capacity is very important.

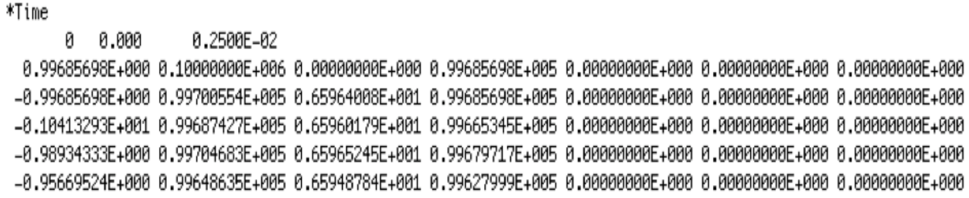
We are considering using a cloud infrastructure managed by Neblina Software [3] in support to these requirements. Neblina is a middleware developed at LNCC that offers users an interface to cloud resources.

4. Challenges

4.1. Data representation

Simulation data can be interpreted as composed by two sets of variables: dimensional and physical quantities. The former places a simulation state into a reference coordinate, whereas the latter informs about the computed physical quantities on each reference point. Typically, the dimensional variables include space and time.

The space dimension refers to a mesh, which represents the topology of the physical domain as a composition of simple geometric objects (eg. a tetrahedron). A mesh is represented by a set of points, referring to the vertices of the geometric objects and a set of edges linking the points and the faces of the model. Observe, yet, that simulations may adopt different scales throughout the domain. Scale in a numeric simulation context conceptually introduces multiple worlds. Given a scale, a world is defined with a specific dimensional system and the set of multidimensional objects visible at that scale with their own schema (i.e. set of physical quantities definitions).



```
*Time
     0   0.000      0.2500E-02
  0.99685698E+000 0.10000000E+006 0.00000000E+000 0.99685698E+005 0.00000000E+000 0.00000000E+000 0.00000000E+000
 -0.99685698E+000 0.99700554E+005 0.65964008E+001 0.99685698E+005 0.00000000E+000 0.00000000E+000 0.00000000E+000
 -0.10413293E+001 0.99687427E+005 0.65960179E+001 0.99665345E+005 0.00000000E+000 0.00000000E+000 0.00000000E+000
 -0.98934333E+000 0.99704683E+005 0.65965245E+001 0.99679717E+005 0.00000000E+000 0.00000000E+000 0.00000000E+000
 -0.95669524E+000 0.99648635E+005 0.65948784E+001 0.99627999E+005 0.00000000E+000 0.00000000E+000 0.00000000E+000
```

Figure 2. Dataout.txt: output file generated by the numerical solver

Furthermore, given a physical quantity, its value in a reference coordinate may change through scales. Finally, a given simulation may be composed of data in different scales, according to the precision requirements in different parts of the physical domain.

In order to illustrate, Fig. 2 shows the first lines of the output file of a 3D numerical simulation. Each line represents the data, computed by a numerical solver, in a mesh point for a given time step. The simulation computes the physical conditions on an artery of the human cardiovascular system. The computed physical quantities are: velocity vectors in three dimensions, pressure, and displacement vectors in three dimensions, respectively.

4.2. Using SciDB for storing simulation data

Our first effort to represent numerical simulation data uses SciDB. The multidimensional array structure is the basis of data representation in SciDB. A user specifies multidimensional structures by providing the range values for each dimension and a list of attribute values to compose a cell. In addition, a versioning mechanism keeps historical values for each attribute. In this context, the following mapping strategy has been defined: *i)* for each set of physical quantities corresponding to a phenomenon being simulated in a given scale; *ii)* define the set of Δ dimensions involved; *iii)* specify the list of physical quantities Π to be computed; *iv)* create an array having the dimensions as Δ and attributes as Π.

In order to illustrate, consider the artery representation as in Fig. 3a and Fig. 3b. The former is a 1D representation of an artery whereas the latter is a 3D model. Assuming that each cross-section of an artery contains 7000 different points in a mesh of the 3D model (Fig. 3b), and the existence of 36000 cross-sections.

Using the AQL (Array Query Language) [12], we would define the following schema for the 3D model of the the artery in Fig. 3b.

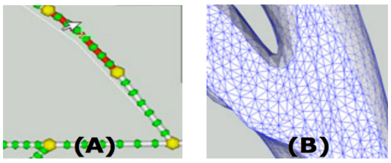


Figure 3. The mesh representation for the models: (a) 1D; and (b) 3D

```
CREATE ARRAY Geometry3D <velocity: point3D, pression: double,
displacement: point3D> [ simulations=0:*,1,0, t=0:500,500,0,
x=1:7000,1000,0, y=1:7000,1000,0, z=1:36000,1000,0]
```

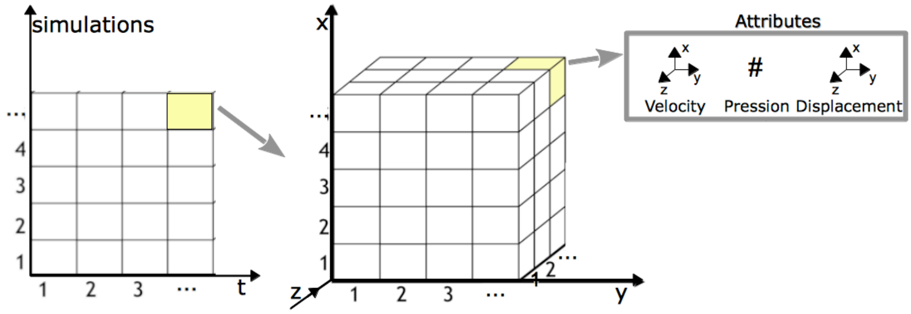


Figure 4. 3D graphical representation of a multidimensional array for the artery model in Fig. 3b

The datatype point3D represents an User Data Type with three variables: values for the vectors in three dimensions in a certain mesh point. The attributes declared between the symbols '<' and '>', represent the physical quantities, while the ones declared between square brackets represent the dimensions of the data schema. For each dimension three values are specified. The first value defines the dimension range, the second specifies the partitioning criteria and the third defines an overlapping range.

Through the AQL query language, we can use the following schema to represent 1D (i.e different scale) of the human cardiovascular system:

```
CREATE ARRAY Geometry1D <velocity: double, pression: double,
flow: double> [ simulations=0:*,1,0, t=0:500,500,0 ]
```

A 1D model represents each cross-section summarized in a single point, as in Fig. 3a. Thus, each point contains the values for the physical quantities on each time step. Additionally, It's refers to the different simulations over the same mesh. Observe that the array data model adopted by SciDB enables the direct representation of multidimensional objects produced in numerical simulations.

4.3. Analyzing numerical simulation data

In order to support the analysis of numerical simulation output, a set of algebraic operators must be provided. We have compiled an initial list of analysis types that would guide the development of analytical operators.

1. computing the past and future cones, according to [13]: given a reference point on a simulation, list the set of points that were responsible [1] for its calculations;
2. comparing simulations results: given two arrays of simulation data, compute their intrinsic distance, and similar wise taking into account observation data;
3. retrieving the set of values in a reference coordinate: given a space-time coordinate, return the set of points and the values of selected physical quantities;

4. drill down through scales: given a reference coordinate in scale s_i, return the corresponding set of points in scale s_j;

AQL and AFL languages, do not have sufficient mechanisms to support all the analytical queries proposed above. The challenge is to create new operations and functions to bridge this gap, as well as, to coupling them to algebraic operators. The current version of AFL, nevertheless, gives support for a handful of analysis over simulation data:

1. obtain the values for pression where the highest values are achieved: `aggregate(Geometry3D, max(pression));`
2. on which time steps did the pressure values were out of stable limits: `filter(Geometry3D, pression < INF or pression > SUP);`
3. show the evolution in time of pressure average: `aggregate(Geometry3D, avg(pression), t);`

4.4. Analytical data management service

From an architectural point of view, we expect to develop a service to interface with the solvers - producing simulation data - and with scientists - submitting analytical queries to the system. Fig. 5 shows the architecture of this service where the Simulation Data Management Service (SDMS) is responsible for providing such an interface as a cloud service. The SDMS should manage the storage and retrieval of the simulation data making it transparent to scientific applications.

The availability of the SDMS as a cloud service fosters the collaborative use of simulation data among scientists of a same research project and among different projects. Additionally, the adoption of the cloud service approach should avoid unnecessary data transfer as the analyses would be executed close to where data resides. Even when data transfer could be needed, for instance, for a new simulation using a given previously computed state as input - all the filtering of unnecessary data would happen close to the data, minimizing the data actually transferred.

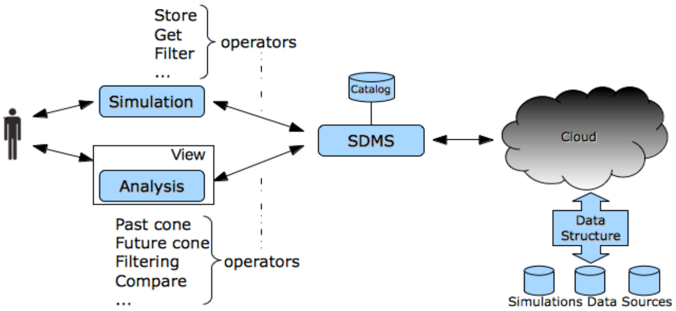


Figure 5. Simulation Data Management Service (SDMS) Architecture

It is important to observe that SDMS would only be used in cases requiring a more detailed tracking of the simulation, as in the debug mode of various IDEs currently available in the market. So, the mechanisms for simulation data storage during the simulation calculus should be activated under discretion, avoiding overloading the system.

Another important aspect of the cloud service approach is the possibility to explore elasticity [8]. Indeed, depending on the requested analysis, a huge amount of data may be retrieved from the data storage device. In such a scenario, the system may allocate extra memory for processing and freeing it as the computation ends. Such a policy is not only a sign of altruism in a collaborative environment, but may reduce the costs involved in supporting the computing platform. Finally, the SDMS should offer to scientific applications an API for accessing its services using traditional programing languages, such as C++, Java an Scripting languages such as Python.

4.5. Scientific computing in cloud

Managing the huge amount of data produced by numerical simulation is a challenge for high performance computing platforms, which are the standard for high demanding scientific applications. Regarding the SDMS, an important issue is the research and development of mechanisms that would enable its deployment in a private cloud as a Service (SaaS) [8]. Additionally, the use of computational clouds becomes relevant due to many other features, such as: scalability; service orientated; flexibility; location transparency; and availability to a large number of users as an appliance. In this context, we should highlight the software Neblina, presented in [3]. Neblina is a middleware developed at LNCC that offers users an interface to cloud resources. Through Neblina a cloud infrastructure, including an application, may be accessed and managed. Typical functionalities include: resources capacity provision, user management, virtualized and physical resources management interface, remote access to the resources and their monitoring.

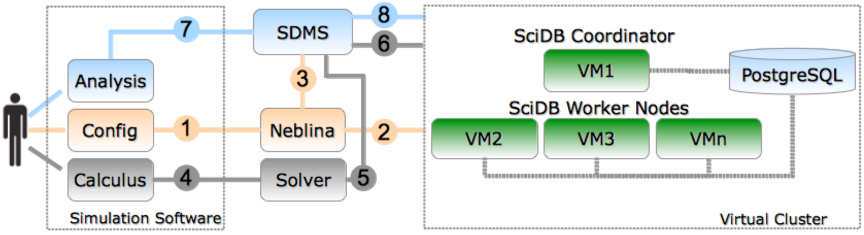


Figure 6. Numerical simulation environment

The SDMS has been integrated into Neblina. This integration makes the cloud environment transparent to SDMS, enabling for instance the activation of its services. In Fig. 6 the architecture of the integrated environment is shown: *1)* First of all, the simulation software interfaces with Neblina in order to create a virtualized environment; *2)* Neblina sets up the environment, creating a virtual cluster to support the simulations data; *3)* Neblina sends resource informations to the SDMS to allow the storage of data generated by numerical solvers; *4)* the simulation software interfaces with numerical solvers to compute the simulation; *5)* numerical solvers requests the SDMS to store the computed results; *6)* the SDMS interfaces with SciDB to store the data; *7)* once the data has been stored, the simulation software can use the SDMS to answer to analytical queries or retrieve simulation data. *8)* finally, the SDMS communicates with SciDB to obtain the requested informations.

5. Conclusion

Numerical simulation is an active area of research wherein scientists model natural phenomenon through computer simulation. The area has benefited from increasingly powerful high performance computing infrastructure to reduce the still long time needed to compute simulations, as much as to enhance the quality of the simulation results. As simulations become more precise, by modeling the phenomenon in higher scale or considering a more fine grained mesh representation of the physical domain, more complex and voluminous becomes the output simulation data. In order to validate scientific hypothesis, the scientist finds himself building complex programs that analyze the simulation output, looking for explanations that may shade some light on the studied phenomenon.

In this paper we investigate the requirements involved in designing a data management service in support for numerical simulation analysis. A multidimensional modeling approach represents the dimensions used in referencing each individual simulation point, and maps each point to its respective physical quantity values. We have identified a set of analytical operations that would leverage numerical simulation results analysis to an analytical level. We discuss the data representation implementation using SciDB. We observe that although the multi-array model adopted by SciDB enables the implementation of the proposed multidimensional representation, further extensions are required to fully support numerical data representation and analysis requirements. In particular, some analysis may require physical quantities to be computed over different abstractions, such as computing their values in a face or edge of a geometry object in a mesh. Moreover, supporting modeling through different scales would require a relationship between multiple representations of the same multidimensional space-time. Some proposed analytical queries can not, as well, be represented using none of the SciDB query languages. New functions and user data types would be needed to cope with those. An initial architecture for a numerical simulation data manager cloud service is provided and its integration with the Neblina cloud middleware is discussed. We expect that this work will provide a better understanding concerning the needs involved in analytical data management for multidimensional numerical simulations.

References

[1] Alexandra Meliou et al. Causality in databases. *IEEE Data Engineering Bulletin*, 33(3):59–67, 2010.

[2] E. Ogasawara et al. An algebraic approach for data-centric scientific workflows. In *37th Intl Conf. on VLDB*, volume 4, pages 1328–1339, Seattle, USA, Aug 2011.

[3] Felipe Fernandes et al. Neblina: Espaços virtuais de trabalho para uso em aplicações científicas. In *XIXX SBRC*, pages 965–972, Campo Grande, Brazil, Jun 2011.

[4] Marta Mattoso et al. Towards supporting the life cycle of large scale scientific experiments. *Business Process Integration and Mgmt*, 5(1/2010):79–92, May 2010.

[5] Michael Stonebraker et al. Requirements for science data bases and scidb. In *Conference on Innovative Data Systems Research*, Asilomar, USA, Jan 2009.

[6] Pablo J. Blanco et al. On the potentialities of 3d-1d coupled models in hemodynamics simulations. *Journal of Biomechanics*, 42(7):919–930, Mar 2009.

[7] Phillipe Cudre-Mauroux et al. A Demonstration of SciDB: A Science-Oriented DBMS. In *22th Intl Conference on VLDB*, Lyon, France, Aug 2009.

[8] Qi Zhang et al. Cloud computing: state-of-the-art and research challenges. *Journal of Internet Services and Applications*, 1(1):7–18, May 2010.

[9] Yi Zhang et al. Storing matrices on disk: Theory and practice revisited. In *37th Intl Conference on Very Large Data Bases*, Seatle, USA, Aug 2011.

[10] Fabio Porto e Stefano Spaccapietra. The evolution of conceptual modeling. chapter Data model for scientific models and hypotheses, pages 285–305. Springer-Verlag, Berlin, Germany, 2011.

[11] Laboratório Nacional de Computação Científica. Medicina Assistida por Computação Científica, Mar 2012. `http://macc.lncc.br/`.

[12] SciDB Inc. *SciDB User's Guide*, 2011. `http://www.scidb.org/`.

[13] John F. Sowa. *Knowledge Representation: Logical, Philosophical, and Computational Foundations*. Course Technology, Aug 1999.