Distributed Database Research at COPPE/UFRJ

Marta Mattoso[1], Vanessa Braganholo[2], Alexandre A. B. Lima[1], Leonardo Murta[2]

[1] COPPE, Universidade Federal do Rio de Janeiro, Brazil
{marta, assis}@cos.ufrj.br
[2] Instituto de Computação, Universidade Federal Fluminense, Brazil
{vanessa, leomurta}@ic.uff.br

Abstract. Our group has been working with different aspects of distributed and parallel processing of databases in the relational, object-oriented, and XML data models. Classic techniques for distributed design and query processing in relational database systems have been revisited to address dynamic issues in high performance computing and flexibility challenges of XML documents. More recently, large-scale scientific data combined with process activities management have introduced challenges to the database and software engineering communities, among several other computer science research areas. Regarding scientific data, challenges are the heterogeneous data formats that encompass relational, XML, binary, and flat files. Our group has been addressing these challenges by capitalizing on our extensive experience in distributed data management. Since each scientific experiment tends to produce and manage its own data, in specific formats, with its own activities (and programs), managing large scale distributed data and activities gets difficult as the amount of heterogeneous data grows.

Categories and Subject Descriptors: H. Information Systems [**H.2 Database Management**]: Systems—*Distributed databases*

Keywords: database clusters, distribution, distributed XML query processing, provenance

1. INTRODUCTION

Nowadays, parallel processing is no longer a niche of specific applications in scientific and financial domains. Almost all computing devices, including cell phones, are multi-core. However, new software is required in order to efficiently use such a massive number of processing elements available in current multi-core devices and future exascale computers [Raicu et al. 2008]. One of the ways data intensive applications can take advantage of these massive processing elements is via parallel processing through data distribution. Data distribution has been the focus of several works in literature [Özsu and Valduriez 2011], and some of these works focus on processing queries in parallel, aiming at improving query performance through distribution.

However, data distribution needs a partitioning model (e.g., horizontal and vertical fragmentation) and careful planning to design data partitions. Managing data distribution involves designing physical partition size, logical partitioning, scheduling, and load balancing. For instance, Ailamaki mentioned in her opening keynote at ICDE 2011 the challenges of new memory hierarchy and stated that data partitioning is a way to eliminate contention. In fact, data partitioning is also a key concept behind the successful map/reduce processing model [Dean and Ghemawat 2008]. Data partitioning has been studied in the database community for many years, but these studies have mainly considered processing elements at the scale of hundreds of units of single core processors.

Moreover, it is no news that database researchers are involved in scientific data management. For

Distributed Database Group: http://dgp.cnpq.br/buscaoperacional/detalhegrupo.jsp?grupo=02021035HQY7JS
We would like to thank CNPq, Capes, FAPERJ, and INRIA for the financial support to our research group.

instance, the Scientific and Statistical Database Management Conference is on its 23^{rd} edition. One should also mention the pioneering work of Medeiros *et al.* [Medeiros et al. 1995], who has evolved into the WOODSS system [Medeiros et al. 2005]. However, more recently, several leading database researchers are focusing on the challenges of data management and scientific processes. Successful technologies in business applications, such as database systems and workflow systems, have been revisited to support scientific experiments. As reported in [Hey et al. 2009], Jim Gray comments: "A fourth data-intensive science is emerging. The goal is to have a world in which all of the science literature is online, all the science data is online, and they interoperate with each other". In other words, managing scientific data is becoming fundamental to science evolution. However, as reinforced by Ailamaki [Ailamaki et al. 2010], computer support to scientific data is still limited. This migration from business support to science is a big challenge.

The research carried out in our group addresses a variety of inter-related issues with focus on distributed techniques on data management. Several problems have been investigated, ranging from issues such as how to efficiently store, fragment, and replicate large databases, to how to efficiently process this data in High Performance Computing (HPC) environments. Our research is divided into three main tracks: (i) adaptive data partitioning, (ii) distributed XML query processing, and (iii) provenance management in scientific workflows. The combination of results from the first two research tracks are directly applied in scientific data partitioning and provenance support in the management of the scientific experiment life cycle.

Regarding our first research track (i.e., adaptive data partitioning), data partitioning outside a database management system has to be coupled to the activities that will process them. Moreover, new data partitioning algorithms are needed to cope with the scientific data deluge [Shoshani and Rotem 2009]. Our work contributes to dynamic database partitioning techniques in current multi-core computers and builds upon them to partition scientific data and to optimize workflow parallel execution. We are open to do this for any kind of data that comes around.

Our second research track (i.e., distributed XML query processing) focuses on challenges imposed by the different data models (eg. relational, XML, etc.). Even though distribution aspects in relational databases have been extensively studied [Özsu and Valduriez 2011], multi-core parallel processing environments pose new challenges with respect to dynamic data partitioning and load balancing. Distribution techniques in the XML data model have been addressed in some works, including ours [Andrade et al. 2006], but it is still a hot topic due to the complexity of the data structure and its operations [Kling et al. 2010]. Our current research goals are to improve distributed relational and XML data support in parallel databases and build upon these results to improve scientific data management support. Specifically, we aim to address challenges in the management of distributed resources inherent in the development of science in large scale.

Finally, our third research track (i.e., provenance management in scientific workflows) raised from the fact that state-of-the-art scientific data management systems tend to treat the data separately from the scientific processes, or even without paying attention to parallel processing. More importantly, current support is based on the workflow execution rather than putting it into the context of the experiment life cycle [Mattoso et al. 2010]. All data generated along the experiment life cycle must be registered and inter-related to be queried through provenance systems. For instance, in recent international interdisciplinary workshops and conferences (e.g., the XLDB conference[1]), the key requirements for scientific data management, which cannot be supported by current technology, have been identified. Among them, we can find distributed and parallel workflow execution involving large numbers of distributed processes and large amounts of heterogeneous data, with support for data provenance in order to understand result data. This is one of the computer science research challenges for the next decade.

[1] http://www-conf.slac.stanford.edu/xldb

Several research initiatives are under development that build upon the tools we have developed, as well as open source and commercial products. Our research group has fostered contact with groups related to target applications such as oil and gas [Guerra et al. 2009; Ogasawara et al. 2009; Oliveira et al. 2009], business [Furtado et al. 2008; Paes et al. 2008; Paes et al. 2009; Lima et al. 2009], and bioinformatics [Cavalcanti et al. 2005; Dávila et al. 2008; Cruz et al. 2010; Gadelha et al. 2011]. Collaborative research activities within these areas have produced several prototypes and new techniques that are currently being used by our partners: Chiron [Ogasawara et al. 2011]; Hydra [Ogasawara et al. 2009; Coutinho et al. 2010]; GExpLine [Oliveira et al. 2010]; ProvManager [Marinho et al. 2010]; SciCumulus [Oliveira et al. 2010]; Heracles [Dias et al. 2010]; Pargres [Mattoso et al. 2005]; Matrioshka [Cruz et al. 2010] and ARAXA [Ferraz et al. 2010].

The purpose of this paper is to briefly survey the main achievements of the Distributed Database Group in the last decade and to discuss challenges to come. Mostly, we will reference our own published work to emphasize our contribution. The rest of this paper is organized as follows. In Section 2, we trace the history of our group. Sections 3, 4, and 5 provide detail on each of our three main research tracks. Section 6 describes some of those techniques developed by the group that were used in real applications. Finally, Section 7 looks towards the future and discusses some challenges in scientific data and workflow management, while Section 8 acknowledges our research partners and sponsors.

2. THE DISTRIBUTED DATABASE GROUP

Our group was created in 1994 at the Computer Science Department of COPPE graduate school of engineering at the Federal University of Rio de Janeiro. It is part of the database research group led by Jano Moreira de Souza since 1986. Marta Mattoso leads the distributed database group with a dynamic and highly collaborative composition. The diversity and large number of researchers participating in co-authored papers of the group shows how effective this cooperation has been. In its current shape, it has a strong collaboration with the Computing Institute at UFF, the Fluminense Federal University. Its main activities are found in Web pages of research projects mainly funded by CNPq, CAPES, FAPERJ, and INRIA (GExp, Dataluge, Sarava, among others), and at the CNPq portal for research groups.

Collaboration with INRIA along the last decade with the research director Patrick Valduriez has evolved into promoting our group to an INRIA Associate Team (*Equipe Associé*[2]). It received the best paper award at the Brazilian-French Workshop COLIBRI (Colloquium of Computation: Brazil/INRIA, Cooperations, Advances and Challenges[3]).

Some of our HPC efforts have led to collaboration with Ian Foster and his group at Argonne Laboratory and University of Chicago. In this joint research effort, we had access to the Teragrid infrastructure [Meyer et al. 2006] and participated on provenance management in HPC through Swift [Gadelha et al. 2010; Gadelha et al. 2011; Gadelha et al. 2011].

All of our scientific workflow contributions are independent of a specific Scientific Workflow Management System (SWfMS). However, most of our tests and evaluations use the VisTrails system[4] as the workflow execution engine. VisTrails has many advantages towards its SWfMS competitors including visualization, version control, and provenance support. Our skills in HPC have led to collaboration with Juliana Freire at the New York University (formerly at University of Utah). Our joint paper [Chirigati et al. 2009] has received the best poster paper award at SBBD 2009.

In this last decade, the group produced 20 journal papers, 6 book chapters, more than 100 complete conference papers, 3 post-docs, 34 MSc and 9 PhD, with 6 ongoing PhD students. In the next four sections we overview the main results we have obtained in our research so far.

[2] http://www-sop.inria.fr/teams/zenith/pmwiki/pmwiki.php/International/Sarava

[3] http://gppd.inf.ufrgs.br/colibri/index.en.html

[4] http://www.vistrails.org/

3. ADAPTIVE DATA PARTITIONING

On-Line Analytical Processing (OLAP) applications typically access large databases using heavyweight read-intensive queries. These queries characterize typical activities in knowledge discovery processes and are usually *ad-hoc*. They can largely benefit from parallel processing techniques. Due to its heavyweight nature, even one single query can be significantly improved through parallelism. The attractive characteristics of DBC (i.e., scalability, reliability, and reduced cost) led us to investigate its use for OLAP query processing and this section describes the main results we achieved.

The *ad-hoc* nature of OLAP queries makes it hard for the database designer to produce a physical data-partitioning schema that can benefit most frequent queries. A more flexible partitioning strategy is virtual partitioning. This flexibility is a key concept in database clusters (DBC), where clusters of PC servers appear as a cost-effective alternative to parallel database servers. Over the last decade, the DBC approach has gained much interest for various database applications. A DBC is a set of PC servers interconnected by a dedicated high-speed network, each one having its own processor(s) and hard disk(s), and running an off-the-shelf DBMS [Akal et al. 2002]. Each cluster node can simply run an inexpensive (non-parallel) DBMS, and so the DBMS is used as a "black-box" component. In other words, its source code is not available and cannot be changed or extended to be "cluster-aware". All extra functionality, like parallel query processing capabilities must be implemented via middleware.

3.1 Virtual Partitioning

Akal *et al.* (2002) proposed a virtual partitioning approach, which we refer to as simple virtual partitioning (SVP). This approach consists in fully replicating a database along a set of sites and breaking each query in sub-queries by adding range predicates. Each DBMS receives a sub-query and is forced to process a different subset ("virtual partition”) of the data. As an example, let us take a relation $R(A, B, C, D)$ and the query Q: SELECT count(*) FROM R. By using SVP, an attribute is chosen as the virtual partitioning attribute. Let us assume A is the VP attribute and it has values in the interval $[a_1, a_m[$. Also, the query will be processed by a DBC with n nodes. Then, SVP produces exactly n sub-queries Q_i, i=1,..., n, of the form:

Q_i: SELECT count(*) FROM R WHERE $A \geq v_i$ and $A < v_{i+1}$

where $v_i = a_1 + (i - 1) \times S$ and $S = (a_m - a_1)/n$. It is important to notice that S is the size of each partition, which for SVP is always the same. The intervals are then calculated and each Q_i is submitted to a different cluster node. One requirement for SVP (and any of its variations) to be efficient is that there must exist a clustered index built on the VP attribute. Such an index guarantees that each cluster node will access a minimum amount of disk blocks.

However, SVP suffers from a major issue: many query optimizers opt for not using an index to access a relation if the estimated amount of tuples to be accessed exceeds a certain threshold. In SVP, the partition size is determined by the number of nodes employed to process a query. So, if such a number is not large enough, the number of tuples in a partition may exceed this threshold and the index may not be used. Load balancing is also an issue. As each node processes only one query, and the DBMS is a black-box component, it is impossible to perform dynamic load balancing for a single query. SVP basic idea is very good due to its flexibility, but needs improvements.

3.2 Adaptive Virtual Partitioning

We address the partition size determination problem in [Lima et al. 2010], originally presented in [Lima et al. 2004], by using an adaptive approach that dynamically tunes partition sizes. We proposed the adaptive virtual partitioning (AVP) approach, which is completely DBMS-independent and uses neither database statistics nor query processing time estimates. AVP avoids full table scans on huge database tables during parallel query processing. It starts just like SVP, producing sub-queries by adding range predicates over the virtual partitioning attribute. Each cluster node receives the same parameterized query and a different interval to process, which is also determined as in SVP. However, instead of executing only one query over the entire interval, the node subdivides the interval and executes many sub-queries. Then, it tries to raise the interval size until there is no performance degradation.

By keeping small intervals, AVP favors the use of clustered indexes. By trying to raise the size of the intervals, it avoids a huge number of sub-queries. To validate our approach, we implemented AVP in a Java prototype and ran TPC-H queries on a 32-node cluster using PostgreSQL. The results show linear and sometimes super-linear speedup for many tested OLAP queries. In the worst cases, almost linear speedup is achieved, which is excellent considering the simplicity of AVP.

The excellent results achieved with AVP lead us to develop a complete database cluster middleware for OLAP query processing, named ParGRES [Mattoso et al. 2005]. Besides solving the partition size problem, AVP also made it possible to provide dynamic load balancing during query processing, which was implemented in ParGRES. The load balancing mechanism is based on *help offering* messages from idle to busy nodes. When a busy node accepts the offer it sends to the idle node part of the interval that still has to be processed. As the database is fully replicated over all nodes, sub-query reallocation becomes easy. Today, ParGRES can be seen as an adaptive map/reduce on top of SQL.

In addition to using TPC-H, we validated our technique by running *ad-hoc* queries over a real world OLAP database (IBGE's Statistical Multidimensional Database - BME), on a 64-node cluster of the Grid'5000 [Bolze et al. 2006] experimental platform. Results are in [Paes et al. 2008]. The experiments showed that, in almost all cases, ParGRES yields super-linear speedup while adding cluster nodes (from 1 to 64 nodes). The experimental results show that AVP improves the performance of *ad-hoc* queries in real scenarios, even with skew.

Then we addressed a major AVP (and ParGRES) issue: the need for full database replication between cluster nodes. Our new strategy combines AVP and Hybrid Design (HD), a table partitioning technique proposed by [Röhm et al. 2000]. From this point, we refer to the original AVP as AVP-FR (AVP with Full Replication). In [Lima et al. 2009], we propose a distributed database design strategy for database clusters based on physical and virtual data partitioning combined with replication techniques. To address the limitation of static physical partitioning, we take advantage of replicas and propose a dynamic query load balancing strategy. In this way, our solution makes it possible to obtain intra-query parallelism during heavyweight query execution with dynamic load balancing while avoiding the overhead of full replication. We call the new approach AVP-PR (AVP with Partial Replication). We ran new experiments with AVP-PR, which achieved excellent speedup [Lima et al. 2009] and decreased disk space utilization.

We also experimented AVP in Grid computing. Ideally, a grid or cloud database solution must respect database autonomy (*i.e.*, avoid database or application migration) while taking advantage of distributed and parallel computing. However, the migration from clusters to grids poses many challenges for parallel query processing, as it must be dealt with in two levels: grid level, which requires the distribution of tasks between grid nodes, and cluster level, which requires the re-distribution of those tasks between cluster nodes (considering the typical scenario where each grid node is a PC cluster). The new level requires attention to load balancing and final result composition.

In [Kotowski et al. 2008], we propose a middleware solution to OLAP query processing in grids, called GParGRES, which employs AVP-FR to provide transparent inter and intra-query processing. Compared with the DBC approach, where the database is replicated at a single site, gAVP-FR enables the database to be replicated at multiple sites, thus increasing data availability and quality of service. We partially implemented the middleware as grid services on Grid'5000. In [Kotowski et al. 2008], we present experimental results obtained with two clusters of Grid'5000 using queries of the TPC-H Benchmark. The results show linear or almost linear speedup in query execution, as more nodes are added in all tested configurations.

$F1_{year} := <CPub, \sigma_{//paper/@date<2000-01-01}>$	$F1_{bib} := <CPub, \pi_{/publication/bibliography}, \{\}>$
$F2_{year} := <CPub, \sigma_{//paper/@date\geq 2000-01-01}>$	$F2_{bib} := <CPub, \pi_{/publication}, \{/publication/bibliography\}>$

Fig. 1. Example of Horizontal (left) and Vertical Fragments (right)

4. DISTRIBUTED XML QUERY PROCESSING

The rise of native XML Database Management System (XDBMS) has made XML query processing more efficient. However, even these XDBMS present poor performance when queries are executed over large amounts of data or when the query is very complex (e.g., *ad-hoc* analytical queries). In general, processing queries over large databases leads to limited performance due to poor memory use and I/O. In fact, [Ferraz et al. 2010] shows experimentally that it is intractable to handle large XML documents in main memory. Inspired by our excellent results with distributed query processing in relational databases, as discussed in Section 3, we have adapted these techniques to the XML model, and shown that performance of XML queries can also be improved by parallelism.

In this section, we discuss the steps we have taken so far in distributed XML query processing, showing the most important results we have obtained.

4.1 Fragmentation Design in XML Data

To be able to obtain good performance by distributing the query processing, the first step is the fragmentation design. In this step, the database is fragmented according to attributes accessed in frequent queries. However, the first challenge to achieve this goal is to decide what an XML fragment is. We thus need a formal definition of a fragmented XML database. Such definition must allow the global collection to be reconstructed from its fragments through reconstruction rules. Such rules are used to decompose the distributed query. Several techniques have been proposed to fragment XML repositories [Bremer and Gertz 2003; Ma and Schewe 2003]. However, they lack some important desiderata for an XML fragmentation model: soundness conditions, the use of an algebra to define the fragments, and support for Single Document (SD) and Multiple Document (MD) collections.

Inspired by the definition of a fragment in relational [Özsu and Valduriez 2011] and object-oriented models [Baião et al. 2004], we have defined three types of fragments [Andrade et al. 2006]: horizontal, where instances are grouped according to a selection predicate; vertical, where subtrees are pruned from the documents by projection operations; and hybrid, where selection and projection are combined. These definitions follow the semantics of the TLC algebra [Paparizos et al. 2004].

As an example, Fig. 1 shows horizontal and vertical fragments that were defined over the collection *CPub*. The horizontal fragments $F1_{year}$ and $F2_{year}$ separate documents containing papers published before January 1^{st} 2001 from those published after that. As an alternative fragmentation design, the vertical fragment $F2_{bib}$ prunes the bibliography subtree from the fragment. This subtree is used as the root of fragment $F1_{bib}$.

Suppose that a collection C (either SD or MD) is decomposed into a set of fragments $\Phi = \{F_1, F_2, ..., F_n\}$. Each fragment is allocated to a node in a distributed environment. We call *distribution design* the definition of fragments and their allocation. For a distribution design to be considered correct, we have also adapted to our context the three correction rules described in [Özsu and Valduriez 2011] that must be satisfied by the fragments [Andrade et al. 2006]: completeness, disjointness, and reconstruction. The completeness rule makes sure that a data item of C is in at least one of the fragments. The disjointness rule makes sure a data item is not in more than one fragment at the same time. Finally, the reconstruction rule determines that it must be possible to reconstruct the original collection from its fragments. For horizontal fragmentation, the union ($\cup$) operator [Jagadish et al. 2001] is used (TLC is an extension of TAX [Jagadish et al. 2001], and for vertical fragmentation, the join ($\bowtie$) operator [Paparizos et al. 2004] is used. We keep an artificial ID attribute in each vertical fragment for reconstruction purposes.

4.2 A Methodology for Distributed Query Processing

Our query processing methodology [Figueiredo et al. 2010] assumes that global XML collections are fragmented and distributed across the network. The methodology involves the decomposition of the main query into sub-queries that will be executed in the remote sites containing fragments of the global collections. Our methodology is an adaptation of the four generic layers proposed by Özsu and Valduriez (2011). Given an XQuery, we first analyze it and transform it into an algebraic query in TLC. This is done by the Query Decomposition layer. Then, the query tree is further analyzed to replace references to the global collection by its corresponding fragments (Localization Layer). The localization layer also reduces the query tree by removing irrelevant fragments (that is, fragments that do not contribute to the query result). Next, Global Optimization is performed. At this point, sub-queries are generated and sent to the local sites. Each site performs Local Optimizations and executes the sub-query. Results are sent back to the central node, consolidated, and finally sent back to the user.

Our experimental results show that queries tend to perform better in the distributed environment. Some queries achieved reductions in the order of 95% when compared to the execution time in the centralized environment. These results are promising, and show that XML can benefit from the parallelism achieved by database fragmentation and distribution. However, they also show that the fragmentation design is very important. We are currently working on this direction (an algorithm to generate the fragmentation design for XML databases).

4.3 Virtual Fragmentation of XML Databases

When queries are *ad-hoc*, such as in scenarios like Decision Support Systems (DSS), physical partitioning design techniques can be an inappropriate solution. In such cases, it is still possible to take advantage of parallelism to improve query performance. The approach we take is to follow the successful virtual partitioning discussed in Section 3 for XML databases. Adapting SVP to the XML model [Rodrigues et al. 2011] was a challenging problem, since the generation of virtual partitions cannot be done by using primary keys (they may not exist in the XML documents). Besides, in contrast with the relational model, the elements and attributes in XML documents are often assigned to string values, which complicates the definition of the intervals that characterize the virtual partitions. Another challenge refers to the types of input data a query can use. In the relational model, SQL queries refer to tables. In XQuery, a query can use specific documents (by using the *doc()* function) or collections (by using the *collection()* function). This difference impacts on the cardinality of elements the query will deal with, and thus impacts on the partitioning algorithm.

To solve the problem caused by the absence of keys, and also to avoid problems with a non-uniform data distribution related to the virtual partitioning attribute, our solution to generate the virtual fragments uses the position of XML elements to define the intervals for the sub-queries. Thus, we modify the original query by adding the selection predicates "*[position()* $\geq v_i$ *and position()* $< v_{i+1}$]" as in SVP. As an example, assume an input document "orders.xml" that contains 1000 orders, and a query that retrieves all orders that were issued before 2001-01-01, as shown in Fig. 2 (left-hand side). Assume also that we want to process this query using 4 cluster nodes. We thus modify the original XQuery to generate 4 sub-queries, each of them ranging over a different interval. Fig. 2 shows the first sub-query, that ranges over the interval [1, 250[.

We have adapted our distributed query processing methodology [Figueiredo et al. 2010] to support *ad-hoc* queries by using the virtual partition technique. The experimental results show that our approach is effective and can reach speed up of up to twenty two times in high cost queries when compared to the centralized environment [Rodrigues et al. 2011]. Our next step is to apply the

```
for $o in doc('order.xml')//order
where $o/date < 2001-01-01
return $o
```

```
for $o in
   doc('order.xml')//order[position() >= 1 and position() < 250]
where $o/date < 2001-01-01 return $o
```

Fig. 2. Original query and one of its sub-queries

adaptive virtual partitioning technique to XML, and see if we can obtain even better results.

5. PROVENANCE MANAGEMENT IN THE LIFE CYCLE OF SCIENTIFIC EXPERIMENTS

Modern scientific investigations depend on huge amounts of data and complex scientific apparatuses. Such investigations, also known as large-scale science, enable a new type of collaborative interdisciplinary research based on shared expertise, instruments, and computing resources. However, these computing resources are often driven by the execution of scripts (a.k.a. pipelines), which specify the sequence of programs that should run to enact a given scientific experiment.

The paradigm of scientific workflows introduced a new model of interaction in the research community, which requires richer information than the one captured in the traditional script-based pipelines. Scientific workflows are an attractive alternative to represent those pipelines or script-based applications. In scientific workflows, activities are often programs or services that represent solid algorithms and computational methods. Current Scientific Workflow Management Systems (SWfMS) focuses on managing the execution of scientific workflows. Nevertheless, executing a workflow is only part of a large-scale scientific experiment. In different situations, multiple scientific workflows compose the same large-scale scientific experiment, which is often interweaved with human-dependent manual operations. In this case, multiple SWfMS may be in place due to the idiosyncrasies of each scientific workflow.

Current SWfMS fail to support the scientific experiment as a whole throughout its life cycle. Therefore, the organization of the experiment life cycle support into a set of integrated experimentation technologies represents a research challenge. This research challenge is especially difficult because it relates different computer science fields besides Databases, including Software Engineering during composition, HPC during execution, and Information Visualization during analysis, among others. In addition, supporting the large-scale scientific experiment life cycle fundamentally requires provenance gathering during all phases. Provenance provides historical information about data manipulated in a workflow [Freire et al. 2008]. This historical information tells us how data products were generated, showing their transformation processes from primary input and intermediary data.

According to Freire et al. (2008), provenance can be categorized into prospective and retrospective. Prospective provenance may be defined as a computational specification of activities of a given workflow and corresponds to the steps that must be followed to generate a data product or class of data products. On the other hand, retrospective provenance defines the actual steps taken during the execution of the workflow. Therefore, the management of provenance information provides to the scientists a variety of data analyses, such as data quality, audit trails, and experiment documentation [Simmhan et al. 2005]. However, the integrated management of provenance of a large-scale scientific experiment composed by multiple scientific workflows executed in different SWfMS becomes a very complex and challenging task.

In this section, we present the several initiatives of our group in terms of provenance management in the large-scale scientific experiment life cycle. Initially, we present our model of the large-scale scientific experiment life cycle. After that, we briefly discuss how conceptual workflow definitions are mapped into an executable workflow using GExpLine, our tool for managing scientific experiments based on software product lines [Northrop 2002]. Moreover, we also describe how a recommendation system can aid the reuse of existing scientific workflows parts. Finally, we describe an approach to the management of provenance in the previously mentioned setting, with different SWfMS enacting

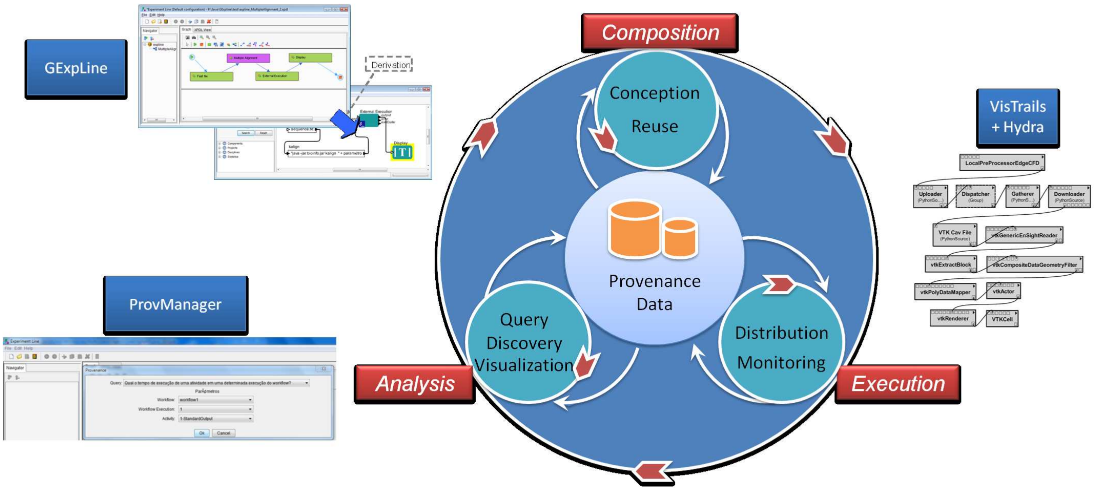

Fig. 3. The proposed scientific experiment life cycle [Mattoso et al., 2010]

different scientific workflows in the same large-scale scientific experiment.

5.1 Our model of a Scientific Experiment Life Cycle

In this section we introduce our model of a scientific experiment life cycle. Fig. 3 presents the large-scale scientific experiment life cycle, which essentially consists of multiple loops traversed by the scientist several times in the course of a scientific experiment. In Fig. 3, the major phases are identified [Mattoso et al. 2010]: *composition*, *execution*, and *analysis*. Each phase has an independent cycle, taking place at distinct moments of the experiment, and handling explicit provenance metadata.

The *composition* phase is responsible for structuring and setting up the whole experiment, establishing the logical sequence of activities, the type of input data and parameters that should be provided, and the type of output data that are generated. The *execution* phase is responsible for materializing the experiment, thus defining exactly the input data and the parameters to be delivered to the SWfMS in order to execute the experiment. Finally, the *analysis* phase is responsible for studying the data generated by the composition and execution phases. A scientist may face two different situations when analyzing the results of an experiment: (i) the result is likely to be correct or (ii) the hypothesis is refuted. In both cases, scientists will probably need new workflow executions to effectively validate the hypothesis or to create a new one. In the case of running several workflows to validate a hypothesis, all the workflows (with different parameters and data sets) must be connected to the same experiment. In fact, most of the existing SWfMS fails to meet this requirement.

5.2 GExpLine support in Workflow Composition using Provenance

The concept of experiment lines [Ogasawara et al. 2009] is an innovative approach to represent a scientific experiment. An experiment line may be defined as a conceptual workflow that is capable of giving rise to multiple concrete workflows. It is a flow of activities where each activity behaves like an independent component [Szyperski 1997]. When an activity of the flow can be implemented by any activity from a list of alternative activities, it is called a variation point. It means that there is more than one alternative program, algorithm, or method to implement the variation point. Also, when an abstract activity can be suppressed when deriving a concrete workflow, in order to represent a different type of result or analysis, and not due to its incorrectness, it is defined as an optional activity. Finally, a mandatory activity is an activity that must be present in all derived concrete workflows. Experiment lines are composed of optional, mandatory, and variant activities.

In addition of supporting experiment lines, GExpLine presents a powerful configuration management mechanism that allows the versioning of workflow elements to model the experiment. In this way, GExpLine allows scientists to [Oliveira et al. 2010]: (i) design experiment lines; (ii) derive concrete workflows; (iii) import concrete workflows from Taverna [Hull et al. 2006], Kepler [Altintas et al. 2004], and VisTrails [Callahan et al. 2006]; (iv) control the versions of abstract/concrete workflows [Ogasawara et al. 2009]; and (v) query prospective provenance data.

5.3 A recommendation system for workflow reuse based on provenance

The conception of a workflow is not a trivial task, and in many cases it becomes a barrier to model more sophisticated experiments. SWfMS like Taverna, Kepler, and VisTrails offer rich graphic interfaces that allow previously registered components to be dragged and dropped into a workflow editing area. However, in current SWfMS, the scope for discovering an activity is limited, since not all components are necessarily registered in the SWfMS. In addition, the knowledge of which activities can be linked to each other is still tacit. It is necessary to run a large number of examples to gain some experience in the setup of the activity flow.

An alternative, often used by scientists, is to try to reuse a previously defined workflow or parts of it. However, the manual reuse is error prone and counterproductive. Aiming to support workflow conception based on reuse, we proposed a recommendation system [Oliveira et al. 2008] that is designed to work over the VisTrails SWfMS. Based on the collaborative filtering approach [Sarwar et al. 2001], prospective provenance is used to recommend to the scientist a set of candidate components to use in the workflow being designed. We infer the need of a component and proactively recommend that component to the scientist. To do so, we parse the previously designed workflow files to extract the relations among components. Those relations are mapped into a database table, containing the components, the port that they are usually connected to, and the workflow itself. Then, each time a scientist adds a component to his current workflow, the tool automatically analyses the database and recommends the most relevant components previously connected to this one, and indicates how they can be connected.

5.4 Data Partitioning in Parallel Workflow Execution

In this section we discuss the challenges for executing scientific workflows in large scale. Our ongoing and future work on supporting data partitioning in parallel workflow execution is similar to a map/reduce approach. A middleware is implemented to bridge the gap between the SWfMS and the high performance computing (HPC) environment. Based on a high level function to partition and distribute scientific data, our parallel workflow activity execution engine allocates data partitions to activities replicated in multiple processors. It follows a kind of bag-of-tasks parallelism, but uses a dynamic data partitioning and performs result composition similar to the reduce part of map/reduce. However, the main contributions of our approach is more than a map/reduce parallelism. It aims at helping the scientist in: (i) identifying data parallelism in workflow activities at an abstract level, (ii) modeling workflow activities using MTC paradigm, (iii) submitting activities from the SWfMS to the distributed environment, (iv) steering by finding failures, detecting performance bottlenecks, monitoring processes status to make the SWfMS aware of the remote execution, (v) gathering prospective and remote retrospective provenance data, and (vi) querying provenance data during and after workflow execution. Despite the advances in SWfMS and provenance support, addressing these issues remains a challenge for the next years. In the following paragraphs we summarize how we are approaching some of these needs of the scientist with focus on parallel processing. We give some details on our ongoing work prototypes with focus on different HPC environments. Hydra and Chiron are focused on MTC clusters. Heracles [Dias et al. 2010; Ogasawara et al. 2010] explores P2P techniques to provide fault tolerance mechanisms and dynamic resource management in MTC, while SciCumulus is focused on adaptive algorithms to match the dynamism of Clouds.

Hydra [Ogasawara et al. 2009] is a middleware solution focused in MTC clusters. The parameter sweep parallelization of Hydra has been evaluated in several simulation workflows from the oil domain using equation solvers [Guerra et al. 2009]. Additionally, Hydra's data partitioning has been used in Fasta files for different bioinformatics workflows [Coutinho et al. 2010].

Cloud computing [Oliveira et al. 2010] provides a new dimension to HPC. Clouds have been shown to be applicable to a wide-range of problems in several domains, including scientific ones [Hey et al. 2009]. An important advantage of clouds is that they do not require scientists to assemble expensive computational infra-structure or even configure many pieces of software since images can be pre-configured to be instantiated on demand. However, executing scientific experiments in cloud environments is still an open problem. SciCumulus [Oliveira et al. 2010] is a cloud middleware for running parallel scientific workflows in clouds with coupled provenance support. Differently from the current mainstream, SciCumulus presents mechanisms to manage and configure the environment before workflow execution, i.e., by automatically creating virtual machines (VM) and setting up virtual clusters. SciCumulus protects scientists from the complexity of configuring and managing cloud environments and collects distributed provenance data from the various instantiated VMs. In [Oliveira et al. 2011] we show the potential of public cloud environments such as Amazon EC2 when it comes to executing parameter sweep involving hundreds (up to 667) variations and thousands of data files that produced several gigabytes of data. As future work in clouds we intend to address workflow steering techniques, failure recovery and other cloud challenges such as security, privacy and dynamic configurations.

All these successful experiences in scientific data parallelism and MTC scheduling have led to the construction of Chiron, a data centric workflow parallel execution engine that encompasses the best ideas of Hydra and proposes optimization heuristics for the parallel execution [Ogasawara et al. 2011]. In Chiron, we addressed the problem of workflow optimization by using an algebraic approach. Our approach is inspired by decades of well-founded query optimization models based on relational algebra to abstract query execution plans. Thus, we propose a scientific workflow algebra, where data is uniformly represented by relations and workflow activities are mapped to operators that have data aware semantics. We conducted a thorough validation of our approach using both a real oil exploitation application and synthetic data scenarios. The experiments run in Chiron demonstrate performance improvements of up to 226% compared to an ad-hoc workflow implementation [Ogasawara et al. 2011].

5.5 ProvManager support in managing provenance data

The main focus of ProvManager [Marinho et al. 2010] is to manage prospective and retrospective provenance in distributed environments. It was conceived to work as a central repository that stores all the provenance data generated by an experiment. Before starting an experiment trial, ProvManager analyzes the workflows that are bound to the experiment to extract prospective provenance and to instrument them with special activities that are able to collect retrospective provenance at runtime. All the collected provenance information is stored in a knowledge base, which is enriched with inference rules. After the experiment trial, scientists are able to query provenance in an integrated way, even if different SWfMS were in-place during the experiment execution.

To illustrate ProvManager in action, consider an experiment structure that is composed of two workflows: the first workflow is instantiated in VisTrails and the other one is instantiated in Kepler. The first workflow is composed by three activities that are running on a remote host. In order to capture provenance data from this workflow, the scientist has to first publish it in ProvManager by uploading the workflow specification (in the VisTrails case, a .VT file). At this moment, ProvManager configures the workflow by automatically adding special activities that will be responsible for capturing and publishing provenance data in ProvManager during workflow execution. Finally, at the end of the instrumentation, a new .VT file is returned to the scientist to be reloaded into VisTrails for execution. During both the instrumentation and execution of the workflow, ProvManager captures provenance data from the workflow and publishes this data into the knowledge base, which is implemented as a

Prolog database.

With all the provenance data collected from the experiment, ProvManager makes the experiment analysis process simpler to the scientist because it works as an integrated place for accessing the provenance data, avoiding to scientist the need to visit each system (in our example, Kepler and VisTrails) individually in a distributed execution environment. In addition, ProvManager provides functionalities to help the scientist manipulate the experiment provenance data, such as a high-level provenance query interface and workflow execution monitoring.

6. APPLICATIONS

This section describes application areas where the techniques discussed in the previous sections have been applied. We have experimented our distributed data partitioning techniques using benchmarks as well as in real applications. We briefly describe some of the applications from collaboration initiatives with the Research Center of Petrobras (the largest Brazilian oil company, a world leader in development of advanced technology from deep-water and ultra-deep water oil production), bioinformatics groups at Fiocruz (The Oswaldo Cruz Foundation, one of the world's main public health research institution), and IBGE (The Brazilian Institute of Geography and Statistics).

Deepwater oil exploitation. An important step in oil exploitation is pumping oil from ultra-deepwater from thousand meters up to the surface through tubular structures, called risers. Maintaining and repairing risers under deep water is difficult, costly, and critical for the environment (e.g., to prevent oil spill). Thus, scientists must predict riser fatigue based on complex scientific models and observed data for the risers. Performing risers fatigue analysis requires a complex workflow of data-intensive activities, which may take a very long time to compute. A typical riser's fatigue analysis workflow [Mattoso et al. 2010] takes as input files containing riser information, such as finite element meshes, winds, waves and sea currents, and case studies, and produces result analysis files to be further studied by the scientists. Some activities, e.g., dynamic analysis, are repeated for many different input files, and depending on the mesh refinements and other riser's information, each single execution may take hours to complete. We have used our tools to support the life cycle of this workflow, from conception through analysis. Most notably, we have successfully used Hydra to obtain near-linear speed-up on the execution of this workflow.

Bioinformatics. Most bioinformatics workflows are data intensive and can largely benefit from data parallelism. However, scientists more concerned with biological results than with parallel programming techniques must expend a large amount of effort to exploit it. By using Hydra's components, data parallelism can be achieved. In [Coutinho et al.], we explored data parallelism in Hydra by modeling and parallel executing a bioinformatics workflow for the identification of orthologous genes in protozoan genomes, which consists of two main activities: BLAST and MCL. We setup a fragmentation cartridge based on FASTA format. Our experiment was implemented on top of the VisTrails SWfMS and the MTC layer was deployed on a SGI Altix cluster of NACAD center at COPPE. Considering the performance of the parallelizable workflow tasks, we achieved a speedup of 99.51 with 128 cores. Based on provenance data collected by Hydra the scientist can fine-tune the experiments.

BME. We have used ParGRES to run typical OLAP queries over the Brazilian official census database (Population and Housing Census 2000) [Paes et al. 2008]. In this application, we used a 64-node cluster of the Grid'5000 platform from INRIA. Our experiments explored intra-query parallel processing with time consuming typical BME census queries. The results showed that, in almost all cases, ParGRES yields super-linear speedup while adding cluster nodes (from 1 to 64 nodes). Our experimental results show that AVP improves the performance of ad-hoc queries in real scenarios, including data skew.

7. CHALLENGES IN SCIENTIFIC DATA AND WORKFLOW MANAGEMENT

Unifying data and the process that generated these data into the same management system is necessary to understand the complexity of an application and to support a richer decision making. The development of systems that use both data and models is still an open issue (VLDB 2011 Challenges and Visions track), even though it was on the agenda of the Asilomar Report (1998). In scientific applications, it is impossible to analyze data separated from their models. SWfMS integrated to provenance systems are working towards this goal, but current solutions are still away from delivering tools that could accelerate the rate of scientific progress. Challenges identified in [Deelman and Gil 2006] are still open, including supporting scientists in the experiment life cycle, i.e., creating, merging, executing, steering, re-using and analyzing the scientific processes.

The very large scale of scientific data management makes scientific workflow management difficult. Some workflow activities may access or produce huge amounts of data and demand HPC environments with highly distributed data sources and computing resources. However, combining SWfMS with such environments to improve throughput and performance through parallelism remains a difficult challenge. In particular, existing workflow development and computing environments have limited support for data parallelism patterns. Such a limitation makes it difficult to automate and efficiently perform parallel execution on large sets of data, which may significantly slow down the execution of a workflow. Some workflows may have to be executed partly on HPC environments and partly on the scientists' own environments. Another difficulty is to support distributed data provenance [Cruz et al. 2009], as described in Section 5, a key function that records critical metadata about experiments (input datasets, parameters, processes, etc.) to help a scientist understand experimental results or reuse existing workflows or parts thereof.

These problems are not addressed by current SWfMS and related research. Current solutions occupy extreme points of a spectrum. Either the SWfMS is semantically rich and provides graphical interfaces, without supporting high performance processing, or the SWfMS is completely dedicated to parallel execution in grids, with no visual interface and low semantic support. Swift, for example, allows scientists to specify parallel workflows using a scripting language. Similarly, MapReduce implementations, such as Hadoop [Wang et al. 2009], allow single activities to be parallelized. These solutions, however, require scientists to stipulate low level parallelization strategies that limit the opportunities for automatic optimizations. Besides the incipient support of SWfMS for parallel processing of workflow activities, there are the new challenges brought by the HPC environments. In 2008, Raicu, Foster and Zhao [Raicu et al. 2008] introduced the Many-Task-Computing (MTC) computational model as a way to represent the large-scale parallelism of tasks that is not supported by current solutions. The challenges regarding MTC are related to the execution of multiple tasks in the context of complex scientific workflows with thousands of processing nodes in clusters, grids or clouds [Oliveira et al. 2010; Dias et al. 2010; Oliveira et al. 2011].

8. ACKNOWLEDGEMENTS

We would like to thank the High Performance Computing Center (NACAD-COPPE/UFRJ) and Grid'5000 (INRIA) where the experiments were performed. We are also grateful to co-authors from local and international groups. Thanks to our international partners, such as, Globus/Swift group: Ian Foster (Univ. Chicago) and Mike Wilde (Argonne Lab); VisTrails group: Juliana Freire, Claudio Silva, and Emmanuelle (Univ. Utah); Atlas group: Patrick Valduriez (INRIA) and Esther Pacitti (Univ. Montpellier). We also thank our colleagues from Rio de Janeiro/Brazil institutions, such as Claudia Werner (COPPE), Guilherme Travassos (COPPE), Fabio Porto (LNCC), Fernanda Baião (UNIRIO) and Maria Luiza M. Campos (PPGI/UFRJ).

Finally, we would like to thank all the students we formed in our group, and list the ones that are currently under our supervision: Anderson Marinho, Carla Rodrigues, Carlos Paulino, Daniel de

Oliveira, Eduardo Ogasawara, Fernando Seabra, Jonas Dias, Luiz Gadelha Jr., Victor Gamboa, and Vitor Silva.

REFERENCES

Ailamaki, A., Kantere, V., and Dash, D. Managing scientific data. *Communications of the ACM* 53 (6): 68–78, 2010.

Akal, F., Böhm, K., and Schek, H. OLAP query evaluation in a database cluster: A performance study on Intra-Query parallelism. In *European Conference on Advances in Databases and Information Systems*. Bratislava, Slovakia, pp. 181–184, 2002.

Altintas, I., Berkley, C., Jaeger, E., Jones, M., Ludascher, B., and Mock, S. Kepler: an extensible system for design and execution of scientific workflows. In *International Conference on Scientific and Statistical Database Management*. Santorini Island, Greece, pp. 423–424, 2004.

Andrade, A., Ruberg, G., Baião, F., Braganholo, V., and Mattoso, M. Efficiently processing XML queries over fragmented repositories with PartiX. In *International Workshop on Database Technologies for Handling XML Information on the Web*. pp. 150–163, 2006.

Baião, F., Mattoso, M., and Zaverucha, G. A distribution design methodology for object DBMS. *Distributed and Parallel Databases* 16 (1): 45–90, 2004.

Bolze, R., Cappello, F., Caron, E., Daydé, M., Desprez, F., Jeannot, E., Jégou, Y., Lanteri, S., Leduc, J., Melab, N., Mornet, G., Namyst, R., Primet, P., Quetier, B., Richard, O., Talbi, E., and Touche, I. Grid'5000: A large scale and highly reconfigurable experimental grid testbed. *International Journal of High Performance Computing Applications* 20 (4): 481 –494, 2006.

Bremer, J. and Gertz, M. On distributing XML repositories. In *International Workshop on the Web and Databases*. San Diego, USA, pp. 73–78, 2003.

Callahan, S. P., Freire, J., Santos, E., Scheidegger, C. E., Silva, C. T., and Vo, H. T. VisTrails: visualization meets data management. In *International Conference on Management of Data*. Chicago, USA, pp. 745–747, 2006.

Cavalcanti, M. C., Targino, R., Baião, F., Rössle, S. C., Bisch, P. M., Pires, P. F., Campos, M. L. M., and Mattoso, M. Managing structural genomic workflows using web services. *Data & Knowledge Engineering* 53 (1): 45–74, 2005.

Chirigati, F., Dahis, R., Cruz, S. M. S. d., Freire, J., Silva, C., and Mattoso, M. L. Q. Desenvolvimento de estruturas de controle explícito para o SGWfC VisTrails. In *SBBD Posteres*. Fortaleza, Brazil, pp. 17–20, 2009.

Coutinho, F., Ogasawara, E., Oliveira, D., Braganholo, V., Lima, A. A. B., Dávila, A. M. R., and Mattoso, M. Many task computing for orthologous genes identification in protozoan genomes using hydra. *Concurrency and Computation: Practice and Experience*. (in press).

Coutinho, F., Ogasawara, E., Oliveira, D., Braganholo, V., Lima, A. A. B., Dávila, A. M. R., and Mattoso, M. Data parallelism in bioinformatics workflows using hydra. In *Emerging Computational Methods for the Life Sciences Workshop*. Chicago, USA, pp. 507–515, 2010.

Cruz, S., Batista, V., Silva, E., Tosta, F., Vilela, C., Cuadrat, R., Tschoeke, D., Davila, A., Campos, M. L. M., and Mattoso, M. Detecting distant homologies on protozoans metabolic pathways using scientific workflows. *International Journal of Data Mining and Bioinformatics* 4 (3): 256–280, 2010.

Cruz, S. M. S. d., Campos, M., and Mattoso, M. Towards a taxonomy of provenance in scientific workflow management systems. In *International Workshop on Scientific Workflows*. Los Angeles, USA, pp. 259–266, 2009.

Dávila, A. M. R., Mendes, P. N., Wagner, G., Tschoeke, D. A., Cuadrat, R. R. C., Liberman, F., Matos, L., Satake, T., Ocaña, K. A. C. S., Triana, O., Cruz, S. M. S., Jucá, H. C. L., Cury, J. C., Silva, F. N., Geronimo, G. A., Ruiz, M., Ruback, E., Silva, F. P., Probst, C. M., Grisard, E. C., Krieger, M. A., Goldenberg, S., Cavalcanti, M. C. R., Moraes, M. O., Campos, M. L. M., and Mattoso, M. ProtozoaDB: dynamic visualization and exploration of protozoan genomes. *Nucleic Acids Research* 36 (1): D547–D552, 2008.

Dean, J. and Ghemawat, S. MapReduce: simplified data processing on large clusters. *Commununications of the ACM* 51 (1): 107–113, 2008.

Deelman, E. and Gil, Y. Final report of the NSF workshop on challenges of scientific workflows, 2006. Available at http://www.isi.edu/nsf-workflows06.

Dias, J., Ogasawara, E., Oliveira, D., Pacitti, E., and Mattoso, M. Improving Many-Task computing in scientific workflows using P2P techniques. In *Workshop on Many-Task Computing on Grids and Supercomputers*. New Orleans, USA, pp. 31–40, 2010.

Ferraz, C. A., Braganholo, V., and Mattoso, M. ARAXA: storing and managing active XML documents. *Web Semantics: Science, Services and Agents on the World Wide Web* 8 (2-3): 209–224, 2010.

Figueiredo, G., Braganholo, V., and Mattoso, M. Processing queries over distributed XML databases. *Journal of Information and Data Management* 1 (3): 455–470, 2010.

Freire, J., Koop, D., Santos, E., and Silva, C. T. Provenance for computational tasks: A survey. *Computing in Science and Engineering* 10 (3): 11–21, 2008.

Furtado, C., Lima, A., Pacitti, E., Valduriez, P., and Mattoso, M. Adaptive hybrid partitioning for OLAP query processing in a database cluster. *International Journal of High Performance Computing and Networking* 5 (4): 251–262, 2008.

Gadelha, L., Mattoso, M., Wilde, M., and Foster, I. Provenance query patterns for Many-Task scientific computing. In *Workshop on the Theory and Practice of Provenance*. Heraklion, Greece, pp. 1–6, 2011.

Gadelha, L. M., Clifford, B., Mattoso, M., Wilde, M., and Foster, I. Provenance management in swift. *Future Generation Computer Systems* 27 (6): 775–780, 2011.

Gadelha, L. M. R., Mattoso, M., Wilde, M., and Foster, I. Towards a threat model for provenance in e-Science. In *Provenance and Annotation of Data and Processes*. Troy, NY, USA, pp. 277–279, 2010.

Guerra, G., Rochinha, F., Elias, R., Coutinho, A., Braganholo, V., Oliveira, D. d., Ogasawara, E., Chirigati, F., and Mattoso, M. Scientific workflow management system applied to uncertainty quantification in large eddy simulation. In *Congresso Ibero Americano de Métodos Computacionais em Engenharia*. Búzios, Brazil, pp. 1–14, 2009.

Hey, T., Tansley, S., and Tolle, K. *The Fourth Paradigm: Data-Intensive Scientific Discovery*. Microsoft Research, 2009.

Hull, D., Wolstencroft, K., Stevens, R., Goble, C., Pocock, M. R., Li, P., and Oinn, T. Taverna: a tool for building and running workflows of services. *Nucleic Acids Research* 34 (2): 729–732, 2006.

Jagadish, H. V., Lakshmanan, L. V. S., Srivastava, D., and Thompson, K. TAX: a tree algebra for XML. In *International Workshop on Database Programming Languages*. Frascati, Italy, pp. 149–164, 2001.

Kling, P., Özsu, M. T., and Daudjee, K. Generating efficient execution plans for vertically partitioned XML databases. *PVLDB* 4 (1): 1–11, 2010.

Kotowski, N., Lima, A. A. B., Pacitti, E., Valduriez, P., and Mattoso, M. Parallel query processing for OLAP in grids. *Concurrency and Computation: Practice and Experience* 20 (17): 2039–2048, 2008.

Lima, A., Mattoso, M., and Valduriez, P. Adaptive virtual partitioning for OLAP query processing in a database cluster. In *Simpósio Brasileiro de Banco de Dados*. Brasília, DF, Brazil, pp. 92–105, 2004.

Lima, A., Mattoso, M., and Valduriez, P. Adaptive virtual partitioning for OLAP query processing in a database cluster. *Journal of Information and Data Management* 1 (1): 75–88, 2010.

Lima, A. A., Furtado, C., Valduriez, P., and Mattoso, M. Parallel OLAP query processing in database clusters with data replication. *Distributed and Parallel Databases* 25 (1-2): 97–123, 2009.

Ma, H. and Schewe, K. Fragmentation of XML documents. In *Simpósio Brasileiro de Banco de Dados*. Manaus, Brazil, pp. 200–214, 2003.

Marinho, A., Murta, L., Werner, C., Braganholo, V., Ogasawara, E., Serra, S. M. S., and Mattoso, M. Integrating provenance data from distributed workflow systems with ProvManager. In *Provenance and Annotation of Data and Processes*. Troy, NY, USA, pp. 286–288, 2010.

Mattoso, M., Berriman, G., Lima, A., Baião, F., Braganholo, V., Aveleda, A., Miranda, B., Almentero, B., and Costa, M. ParGRES: a middleware for executing OLAP queries in parallel. Tech. Rep. ES-690, PESC/COPPE/UFRJ, 2005.

Mattoso, M., Werner, C., Travassos, G. H., Braganholo, V., Murta, L., Ogasawara, E., Oliveira, D., Cruz, S. M. S. d., and Martinho, W. Towards supporting the life cycle of large-scale scientific experiments. *International Journal of Business Process Integration and Management* 5 (1): 79–92, 2010.

Medeiros, C., Vossen, G., and Weske, M. WASA: a workflow-based architecture to support scientific database applications. In *Database and Expert Systems Applications*. London, United Kingdom, pp. 574–583, 1995.

Medeiros, C. B., Perez-Alcazar, J., Digiampietri, L., G. Z. Pastorello, J., Santanche, A., Torres, R. S., Madeira, E., and Bacarin, E. WOODSS and the web: annotating and reusing scientific workflows. *SIGMOD Record* 34 (3): 18–23, 2005.

Meyer, L., Annis, J., Wilde, M., Mattoso, M., and Foster, I. Planning spatial workflows to optimize grid performance. In *ACM Symposium on Applied Computing*. New York, USA, pp. 786–790, 2006.

Northrop, L. SEI's software product line tenets. *IEEE Software* 19 (4): 32–40, 2002.

Ogasawara, E., Dias, J., Oliveira, D., Porto, F., Valduriez, P., and Mattoso, M. An algebraic approach for Data-Centric scientific workflows. *PVLDB* 4 (12): 1328–1339, 2011.

Ogasawara, E., Dias, J., Oliveira, D., Rodrigues, C., Pivotto, C., Antas, R., Braganholo, V., Valduriez, P., and Mattoso, M. A P2P approach to many tasks computing for scientific workflows. In *International Meeting on High Performance Computing for Computational Science*. Berkeley, USA, 2010.

Ogasawara, E., Oliveira, D., Chirigati, F., Barbosa, C. E., Elias, R., Braganholo, V., Coutinho, A., and Mattoso, M. Exploring many task computing in scientific workflows. In *Workshop on Many-Task Computing on Grids and Supercomputers*. Portland, Oregon, pp. 1–10, 2009.

Ogasawara, E., Paulino, C., Murta, L., Werner, C., and Mattoso, M. Experiment line: Software reuse in scientific workflows. In *International Conference on Scientific and Statistical Database Management*. New Orleans, USA, pp. 264–272, 2009.

Ogasawara, E., Rangel, P., Murta, L., Werner, C., and Mattoso, M. Comparison and versioning of scientific workflows. In *Workshop on Comparison and Versioning of Software Models*. Vancouver, Canada, pp. 25–30, 2009.

Oliveira, D., Baião, F., and Mattoso, M. Towards a taxonomy for cloud computing from an e-Science perspective. In *Cloud Computing: Principles, Systems and Applications*, Nick Antonopoulos and Lee Gillam ed. Computer Communications and Networks. Springer-Verlag, Heidelberg, pp. 47–62, 2010.

Oliveira, D., Cunha, L., Tomaz, L., Pereira, V., and Mattoso, M. Using ontologies to support deep water oil exploration scientific workflows. In *IEEE International Workshop on Scientific Workflows*. Los Angeles, USA, pp. 364–367, 2009.

Oliveira, D., Ocana, K., Ogasawara, E., Dias, J., Baião, F., and Mattoso, M. A performance evaluation of x-ray crystallography scientific workflow using SciCumulus. In *International Conference on Cloud Computing*. Washington D.C.,USA, pp. 708–715, 2011.

Oliveira, D., Ogasawara, E., Baião, F., and Mattoso, M. SciCumulus: a lightweigth cloud middleware to explore many task computing paradigm in scientific workflows. In *International Conference on Cloud Computing*. Miami, USA, pp. 378–385, 2010.

Oliveira, D., Ogasawara, E., Seabra, F., Silva, V., Murta, L., and Mattoso, M. GExpLine: a tool for supporting experiment composition. In *Provenance and Annotation of Data and Processes*. Troy, NY, USA, pp. 251–259, 2010.

Oliveira, F., Murta, L., Werner, C., and Mattoso, M. Using provenance to improve workflow design. In *Provenance and Annotation of Data and Processes*. Salt Lake City, USA, pp. 136–143, 2008.

Özsu, M. T. and Valduriez, P. *Principles of Distributed Database Systems*. Springer, 2011.

Paes, M., Lima, A., and Mattoso, M. Processamento de alto desempenho em consultas sobre bases de dados geoestatísticos usando replicação parcial. In *Simpósio Brasileiro de Banco de Dados*. Fortaleza, Brazil, pp. 241–255, 2009.

Paes, M., Lima, A. A. B., Valduriez, P., and Mattoso, M. High-Performance query processing of a Real-World OLAP database with ParGRES. In *International Meeting on High Performance Computing for Computational Science*. Toulouse, France, pp. 188–200, 2008.

Paparizos, S., Wu, Y., Lakshmanan, L. V. S., and Jagadish, H. V. Tree logical classes for efficient evaluation of XQuery. In *International Conference on Management of Data*. Paris, France, pp. 71–82, 2004.

Raicu, I., Foster, I., and Zhao, Y. Many-task computing for grids and supercomputers. In *Workshop on Many-Task Computing on Grids and Supercomputers*. Austin, USA, pp. 1–11, 2008.

Rodrigues, C., Braganholo, V., and Mattoso, M. Virtual partitioning in queries over distributed XML databases. *Journal of Information and Data Management*, 2011. (in press).

Röhm, U., Böhm, K., and Schek, H. OLAP query routing and physical design in a database cluster. In *International Conference on Extending Database Technology*. Konstanz, Germany, pp. 254–268, 2000.

Sarwar, B., Karypis, G., Konstan, J., and Reidl, J. Item-based collaborative filtering recommendation algorithms. In *International Conference on World Wide Web*. Hong Kong, China, pp. 285–295, 2001.

Shoshani, A. and Rotem, D. *Scientific Data Management: Challenges, Technology, and Deployment*. Chapman and Hall/CRC, 2009.

Simmhan, Y. L., Plale, B., and Gannon, D. A survey of data provenance in e-science. *SIGMOD Record* 34 (3): 31–36, 2005.

Szyperski, C. *Component Software: Beyond Object-Oriented Programming*. Addison-Wesley Professional, 1997.

Wang, J., Crawl, D., and Altintas, I. Kepler + hadoop: a general architecture facilitating data-intensive applications in scientific workflow systems. In *Workshop on Workflows in Support of Large-Scale Science*. Portland, USA, pp. 12:1–12:8, 2009.