Challenges and Opportunities in Self-Managing Scientific Databases

Thomas Heinis, Miguel Branco, Ioannis Alagiannis, Renata Borovica
Farhan Tauheed, Anastasia Ailamaki

Data-Intensive Applications and Systems Laboratory, École Polytechnique Fédérale de Lausanne, Switzerland
{firstname.lastname}@epfl.ch

Abstract

Advances in observation instruments and abundance of computational power for simulations encourage scientists to gather and produce unprecedented amounts of increasingly complex data. Organizing data automatically to enable efficient and unobstructed access is pivotal for the scientists. Organizing these vast amounts of complex data, however, is particularly difficult for scientists who have little experience in data management; hence they spend considerable amounts of time dealing with data analysis and computing problems rather than answering scientific questions or developing new hypotheses. Therefore scientific experiments are in many ways ideal targets for research in self-managing database systems.

In this paper, we describe challenges and opportunities for research in automating scientific data management. We first discuss the problems faced in particular scientific domains using concrete examples of large-scale applications from neuroscience and high-energy physics. As we will show, the scientific questions are evolving ever more rapidly while datasets size and complexity increases. Scientists struggle to organize and reorganize the data whenever their hypothesis change and therefore their queries and their data changes as well.

We identify research challenges in large-scale scientific data management related to self-management. By addressing these research challenges we can relieve the burden of organizing the data off the scientists, thereby ensuring that they can access it in the most efficient way and ultimately enabling the scientists to focus on their science.

1 Introduction

We are entering the "data deluge" era where scientists are flooded with experimental data [3] because they use ever-faster hardware to run simulations and ever more precise instruments to observe phenomenon in space or nature. While this data is much easier to collect today due to advances in storage systems, our ability to organize and process this data has not kept pace with the growth; and scientists struggle to organize it for fast access. Drawing examples from observational science, the telescopes of the Sloan Digital Sky Survey (SDSS) record 73 TB raw observation data annually, while the instruments of CERN's large hadron collider (LHC) store 15 PB raw event data per year. In the simulation sciences, scientists are also struggling with data deluge. For example,

Copyright 2011 IEEE. Personal use of this material is permitted. However, permission to reprint/republish this material for advertising or promotional purposes or for creating new collective works for resale or redistribution to servers or lists, or to reuse any copyrighted component of this work in other works must be obtained from the IEEE.
Bulletin of the IEEE Computer Society Technical Committee on Data Engineering

already today the detailed models of the neuroscientists in the Blue Brain project (BBP) [2] are several hundreds of Gigabytes big and the simulation output has a size in the order of Petabytes (depending on the simulation length). Models as well as simulation data are predicted to grow by orders or magnitude in the coming years. In addition, while each of the aforementioned projects (SDSS, LHC and BBP) individually forecasts a tremendous growth in data, Gray et al. [3] predict that the amount of scientific data across all projects and disciplines will double every year.

Quick access to these data sets is crucial for the scientists as it helps them to speed up the cycle of experiment preparation, execution, analysis and refinement. Only thorough analysis of the data enables them to improve and refine their simulations or to calibrate their instruments, ultimately helping them to develop and corroborate new hypotheses. Deploying, maintaining and tuning a data management system for their perpetually changing query workloads and data, however, is a tremendously hard problem. The complexity as well as lack of support for complex data models discourage scientists from using DBMSs and often lead them to develop home-grown solutions that neither scale nor are efficient. Scientists are in dire need of scalable tools and methods for automated organization of the data, relieving them from having to do it themselves and ultimately allowing them to concentrate on their science again.

In this paper, we discuss the data management problems scientists in different disciplines face. We analyze how state-of-the-art methods can help scientists to tackle their data issues, but more importantly, we point out the gaps between what data management tools and approaches offer today and what researchers need to move ahead in their disciplines. As we show, these gaps translate into exciting research challenges related to automating data management and we sketch a roadmap for self-managing database management research for scientific data. In particular, we identify the following problems and their data management research challenges:

a) Amount of Data: the rapid growth of data volumes leads to two problems for scientists. First, having to wait until all data is loaded into the database system before they can analyze it keeps them from using database systems at all. Research into querying raw data files is hence needed so they can start querying their data instantly. Second, scaling into the cloud is a straightforward way to deal with the data volumes. Organizing the data in the cloud efficiently (partitioning, distributing, etc.), however, complicates the automated physical design problem, requiring more research in this area.

b) Fast-Changing Environment: many scientists constantly revise their hypothesis and hence they collect different data over time and they also continuously change the queries asked. Consequently, finding the proper physical design of the data is no longer a one-off operation, but instead has to be done continuously, challenging today's automated physical design methods.

c) Fine-grained Models and Complex Queries: the models used in simulation sciences grow equally complex. Today's indexes do not scale to the level of detail of their spatial models. Similarly, the questions scientists today ask about their data have become increasingly complex. No indexes exist to answer them efficiently and so scientists resort to using unscalable workarounds. New indexes are needed to answer the questions efficiently and thus add to the complexity of the automated physical design problem.

The rest of this paper is structured as follows: in the first part in Section 2 we identify data management problems in high energy physics and in neuroscience. In response to these problems, we identify the challenges involved in developing self-managing database systems for scientists in the second part in Section 3. We conclude in Section 4.

2 Data Management Problems in the Sciences

Data management challenges vary across different scientific disciplines. In the following we discuss the problems in two example disciplines, high energy physics from the observational sciences and neuroscience from the

simulation sciences. While in the former the problems are mostly related to the sheer size of the data resulting from experiments and the challenges are related to large-scale data organization, the challenges in the latter arise primarily from the growing complexity of the data and the questions asked about it.

2.1 High-Energy Physics

High-Energy Physics studies the fundamental constituents of matter and the laws governing their behavior. It relies on high energy probes or particle accelerators to break particles into their building blocks. The results are observed by large detectors and are analyzed to validate and formulate new theories. The ATLAS [4] experiment, one of the detectors of the Large Hadron Collider (LHC), collects tens of Petabytes of raw data per year. This data is being continuously analyzed by a large number of physicists in universities and research institutions worldwide. The data analysis for an experiment like ATLAS is a complex process (see [5] for a simplified view). The events at the end of this process are stored on a permanent basis and are then passed through a series of reconstruction and analysis steps to infer and calculate their properties. Events are also regularly re-analyzed given newer and more precise algorithms.

The main challenges for self-manageability in the LHC context are that (a) there is a need to store, share and analyze Petabytes of data and (b) it is a fast-changing environment, with regular modifications made to the software and hardware, but more importantly, depending on the hypothesis, researchers want to corroborate changes to queries and data. In the remainder of this section we focus on the two aforementioned data management challenges. We discuss the complexity of the currently-used storage solution in ATLAS and how the fast-changing environment makes the data management problem even more demanding.

2.1.1 Amount of Data

The LHC generates a total 15 PB of new data per year [6], most of which is raw data acquired by the detectors and is collected on a nearly continuous basis. The remainder is derived data products being produced during processing, simulated data and metadata (accelerator logs, conditions, geometry, calibration or alignment data etc). The derived data is significantly smaller than the raw data. In the case of ATLAS, the derived data products are either the detailed output of an event reconstruction with sufficient information to identify particles or summaries of the reconstructed event with sufficient information for common analysis. Currently the LHC uses a hybrid storage solution for its data, with some data stored in relational databases and the remaining in flat files using in-house data formats (e.g., POOL [7]). Flat files are used to store most of the event data using an object-oriented model, while relational databases contain high-level catalogs.

Just at CERN, one of the largest relational databases is the LHC accelerator logging database with over 50 TB. All relational databases are hosted in 100 Oracle RAC database clusters of 2 to 6 nodes. In total there are over 3000 disk spindles providing just over 3 PB of raw disk space for relational data. The remaining (event) data is stored in storage systems like the CASTOR [8] service, which uses a tape backend for long-term data preservation as well as a distributed file system for data access from batch processing nodes. CASTOR currently hosts approximately 50 PB of data in over 300 million files at CERN. A growth rate of Petabytes per year, data from various experiments, numerous files stored both in relational databases and using custom formats and the need to store and organize produced data in order to provide efficient access to participants from different institutions constitute only some of the parameters in the tremendously hard data management problem scientists face in the ATLAS project. Each of the problems enumerated before would pose significant manageability issues by itself; their combination demands automated systems.

2.1.2 Fast-Changing Environment

The high-energy physics environment and science exploration in general is subject to frequent changes. For instance, analysis algorithms are improved regularly and new features or applications are implemented in response

to scientific needs. The scientific needs and with it the queries and the data change quickly as the phenomenon the scientists study evolves. Each query initiates an exploration step in the data and its result might change the way scientists understand the data and the way they will access the data in future queries. Finding the optimal physical design therefore is not a static problem but a dynamic and ever-evolving process.

The infrastructure also changes at a rapid pace. There are multiple institutions participating, each with its own hardware acquisition and renewal cycles or software upgrades (including security patches). The LHC experiments rely on strict operational procedures to minimize the impact of changes. Integration and pre-production testbeds are used to ensure that changes are tested well in advance. Database administrators (DBA) are responsible for these tasks and any possibility for automation is important to alleviate the heavy workload.

2.2 Neuroscience

For at least two and a half thousand years, humans have tried to understand what it means to perceive, to feel, to remember, to reason and to know. In the last two centuries, the quest to understand the brain has become a major scientific enterprise. The increasing prevalence of brain disease has lent new urgency to this quest. To better understand and answer fundamental questions about the brain, neuroscientists have therefore started to build and simulate models of the brain on an unprecedented level of detail on an equally unparalleled scale.

For this reason, the neuroscientists in the Blue Brain project (BBP) [2] analyze real rat brain tissue in order to simulate an in-silico brain model on a supercomputer (BlueGene/P, 16K cores). Instead of simulating neural networks, the neuroscientists in the BBP already simulate a model at unprecedented levels of detail (with each neuron represented by several thousand cylinders) and now start to build even finer grained models. For instance, to better understand the propagation of impulses between neurons, they model the conductivity of the neurotransmitter at the subcellular level.

The small models used at the outset of the project occupy in the order of GBs and can easily be handled by state-of-the-art access methods. Today, however, neuroscientists build and simulate models of 100,000 or 500,000 neurons, amounting to 350 Gigabytes or 1.7 Terabytes respectively on disk. These sizes can no longer be handled efficiently by state-of-the-art indexes used to build and analyze the models. At the same time, today's models are still orders of magnitude smaller than a model of the entire human brain with 100 Billion neurons, the ultimate goal of the neuroscientists in the BBP. In the following we discuss how the increasingly detailed models as well as the increasingly complex questions challenge state-of-the-art indexing methods.

2.2.1 Fine-grained Models

In addition to scaling the size, neuroscientists also want to build and simulate the brain at ever more fine-grained detail, increasing the level of detail many times. When at the outset of the project each neuron was only modeled with several thousands of cylinders, today several thousand synapses are added to each neuron. The plan is to model the neurotransmitter as well as the organelles of the neurons at the molecular level, leading to several orders of magnitude more detailed/dense model.

Building ever more fine-grained models is also a trend in other sciences (models of the human arterial tree, lung airways, protein synthesis etc.). During many tasks in building and analyzing these models, scientists rely on the efficient execution of spatial queries on the models or on the simulation results. Spatial indexing and data partitioning tools at the scientists disposal, however, do not scale when it comes to analyzing the more detailed models due to the density of spatial datasets. Most state-of-the-art indexes, e.g., the R-Tree [10], build on a hierarchical organization of the spatial data. Like the R-Tree, however, they suffer from the problem of overlap, which slows down query execution considerably. Because overlap grows with increasingly fine-grained models, current index technology needs to develop new classes of spatial indexes which are oblivious to overlap.

2.2.2 Complex Queries

Not only the models, but also the questions the neuroscientists ask have become ever more complex. To analyze the models efficiently, simple spatial queries such as range or point queries are no longer enough to answer sophisticated scientific questions. For instance, the question "What postsynaptic synapses are close to a particular dendrite?" translates into a nearest-neighbor query where the neighbors must satisfy additional constraints.

The neuroscientist's workaround for complex queries is to implement ad-hoc solutions based on basic spatial queries. For the example query, they use a nearest-neighbor approach that retrieves near neighbors until enough neighbors satisfying the additional constraints are found. Clearly this means that many objects are retrieved unnecessarily and that by using the constraints earlier in the execution, the query could be executed substantially faster. As datasets grow exponentially, it is clear that workaround solutions for complex queries do not scale and that new indexes answering complex questions efficiently need to be developed.

3 Research Challenges in Automated Physical Design

Each of the challenges mentioned above (amount of data, fast-changing environments, need for fine-grained models and complex queries) significantly complicates the design of self-managing database systems; addressing all challenges simultaneously appears to be an unattainable goal in the short or even medium term. Nonetheless, in this section we describe promising avenues of research, each tackling part of the problem of developing self-managed databases for the sciences. In particular, we describe the research challenges in dealing with huge amounts of data, with fast evolution of queries and data, with increasingly fine-grained models in the simulation sciences and with complex queries. Addressing the challenges will relieve scientists from dealing with current data management technology, namely from defining a suitable schema, designing a proper physical organization and loading the data inside a DBMS. Because scientists typically work with data in an explorative way, i.e., their next query depends on the last, they need answers instantly and can only rarely rely on offline processing and analysis approaches for big data [21].

3.1 Coping with Large Amounts of Data

In order to support ever-growing data volumes and achieve high performance at the same time, DBMS today need to rely on parallelism [15]. A definite step toward this end is cloud computing, where farms of commodity machines are exploited to provide storage facilities and offer resources as a service. In the following, we discuss the challenge of scaling the data into the Cloud and addressing the associated problem of finding the proper physical design for databases distributed in such an environment. Then, we discuss an approach that bridges the chasm between scientists and database systems, by providing features for querying raw data files in-situ, i.e., without the need for data loading a priori, defining a schema or any other hurdle that repels scientists from using DBMS in the first place.

3.1.1 Scaling into the Cloud

Partitioning the data and distributing it in the cloud is the straightforward way of handling its sheer size. What makes the cloud computing additionally appealing to scientists is that it relieves them from worrying about purchasing the software, provisioning the hardware for their computation, maintaining hardware and software or providing reliability (through data redundancy or disaster recovery). Scientists can outsource their computation to public cloud providers and pay for the service usage on demand, exploiting a *pay-as-you-go* charging principle [14]. By sharing storage and computational capacities, and accordingly energy cost as well, customers are likely to pay much less than in case they buy all the facilities for themselves. Additionally, provided there is mutual trust among scientists, use of the Cloud also facilitates sharing their data and research results among

each other, while leaving the details such as data ownership, data confidentiality and data dependencies to the cloud providers.

Scaling into the cloud however also involves major research challenges, particularly in the area of physical database design. Configuring a DBMS residing on a single machine and choosing a proper physical design that will boost the performance of an a priori given training workload is a difficult task that requires profound knowledge of different physical design structures and their interactions. Configuring it correctly is of paramount importance, since the right physical design can improve performance of the running queries by several orders of magnitude. To replace the need of having a highly skilled and expensive expert, commercial DBMS vendors offer automated physical design tools that perform database tuning automatically [1, 16]. Despite being able to reduce database ownership cost significantly, existing automated physical design tools employ different heuristics to cope with the complexity even when considering a DBMS residing on a single machine [1, 16]. In a Cloud setting, the problem is further exacerbated by having additional parameters that need to be considered, like for example data ownership.

Because data movement operations in the cloud are much more expensive than relational operations on each machine, partitioning the data appropriately becomes a critical design problem. Additionally, in case of scientific applications, data related to the same experiment is often stored distributed on equipment owned by different institutions. Hence, during partitioning or moving operations, the data ownership has to be considered, adding a new dimension to the physical design problem.

By scaling into the cloud, the physical design problem becomes even more difficult because the cloud employs *multi-tenancy* , i.e., data owned by different tenants co-reside in the same database. Multi-tenancy introduces less predictability in the self-tuning process, and opens a new research area that deals with the problems of efficient resource virtualization between different tenants that co-reside on the same physical machine. Furthermore, an additional problem the cloud providers have to cope with now is unpredictability in the workload variations and how those changes affect the performance of the running applications.

Despite introducing considerable challenges in the physical design, the cloud brings significant opportunities for scaling massive amounts of data, which, considering the data deluge problem, becomes the right direction for the future.

3.1.2 Querying Raw Data

For very large experiments such as the LHC, the scientific data is usually stored in a combination of systems, ranging from relational databases to plain files in a distributed file system. Practitioners, particularly in the scientific field, often opt to store large amounts of data outside relational database systems due to concerns regarding scalability, cost or long-term data conservation.

We expect that relational databases will continue to coexist with non-relational data. In fact, it is possible that a smaller percentage of data will be stored in classical relational databases, as seen with the growth in popularity of systems such as Hadoop [21] or NoSQL databases. Part of the reason is that classical database systems can only be used after data is loaded and physical design is performed. These tasks require a significant investment and cause a delay between the time data is collected and the first queries are submitted (data-to-query time). As data collections grow larger and larger, performing the traditional extract/transform/load/query cycle is becoming tedious, impractical and at the extreme, infeasible.

Additionally, it is often not clear what particular share of data scientists need to analyze. Loading entire datasets to discard them after a few queries is an undesirable state of affairs. It is hence no surprise that initialization – along with tuning – is frequently cited as one of the most deterring factors for the adoption of DBMSs. As a result, many communities, especially in the scientific domain, revert to home-grown solutions, which are easier to manage but usually do not employ the sophistication of an advanced DBMS [17].

Nowadays, a DBMS can only be used once the data is entirely loaded into it – a time consuming and often unnecessary process. DBMS have long provided rich functionality on datasets loaded into a database, but only

limited capabilities are offered to query external files of raw data. It has become apparent that the database community needs to reconsider the loading prerequisite and redesign database systems to query data in-situ, i.e., in its original location and original data formats (plain files or relational schemas). The in-situ data processing model can significantly reduce the data-to-query time and lead to user-friendlier systems. Nevertheless, renouncing the benefits of data loading while maintaining the performance of a classic DBMS comes with many challenges such as (a) reduce the extra costs of accessing and analyzing the raw data for every query (b) improve data retrieval in future accesses to the raw data files (c) store the data into the proper format and adjust its access methods according to the incoming queries (d) make all this evolving process as transparent as possible to the user via lightweight adaptation actions (e) provide support for multiple data formats with different characteristics (e.g. HDF-5, NetCDF).

3.2 Coping with Fast-Changing Environments

Fast-changing environments additionally exacerbate the physical design problem. With the goal of proposing an optimal design for a given workload, existing physical design tools require the training workload to be known beforehand. Unfortunately, in the scientific domain, it is usually difficult to obtain a set of training queries a priori. The latter is an indirect consequence of the scientific discovery process, where research directions change frequently based on recently gathered knowledge. Regardless of a performance improvement it may bring, in rapidly evolving scientific repositories the effort of finding the proper design may be worthless, because the researcher's focus may shift and the queries may change from the ones for which the system is tuned.

It is hence evident that continuous, automated and adaptive tuning is important in scientific disciplines. In an ideal scenario, scientists would rely on autonomous techniques such as database cracking [18] to gradually create indexes on commonly accessed data, without any user intervention.

3.3 Coping with Fine-grained Models

With ever-faster hardware and better tools, the models scientists simulate become increasingly complex. The complexity manifests itself in the size of the spatial data sets they use but more importantly in their density (the number of elements per unit of space). As discussed previously, several types of spatial queries can no longer be answered efficiently on increasingly detailed models. Coping with fine-grained models is not just a problem in neuroscience, but affects all simulation sciences where scientists work with models. Examples include the simulation of the human arterial tree in computational fluid dynamics research, earthquakes in geology and protein synthesis in systems biology.

Indexes supporting scientists in the simulation sciences in executing distance joins [9], range queries [10] and others all build on the R-Tree or related tree-like index structures [11]. By building all these approaches on the R-Tree, however, we also inherit its inefficiencies, namely performance deteriorates due to overlap. Overlapping bounding boxes on one level lead to ambiguous paths through the tree. When executing a query, all paths need to be followed and overlap thus leads to an excessive number of pages read from disk. The number of pages read from disk defines the query execution time and hence reading too many pages slows down query execution substantially. Several approaches have been developed to tackle the overlap issue, all of them, however, have drawbacks.

For example, neuroscientists frequently ask spatial range queries to determine what neurons of the brain model are in a particular region. The results of this very basic query is used for scrutinizing interesting parts of the brain, for the purpose of analysis (tissue density), visualization etc. Many approaches to support range queries on spatial data have been proposed. Yet, state of the art approaches (R-Trees and variants [11, 10]) do not scale well for data sets of increasing density because the overlap increases. Several approaches have been developed to tackle the problem of overlap, each of them, however, has drawbacks. The R+-Tree [12] for instance requires excessive disk space.

Today's increasingly dense data sets suffer even worse from overlap, because with the increasing density of the data set also the overlap in the R-Tree increases. Clearly new indexing approaches that scale with increasing data set density must be developed. In addition, new indexing approaches for fine-grained models will further complicate the automated physical design problem. Not only does a decision need to be reached whether or not to use spatial indexes but also what index is suited best for a particular spatial query.

3.4 Coping with Complex Queries

While the models scientists develop become increasingly complex, so do the questions they ask about them. While this is a problem across different disciplines in simulation sciences (computational fluid dynamics, systems biology, etc.), we illustrate it with two particular examples from neuroscience: while the problem of nearest neighbor queries has been studied before, the nearest neighbor queries problem with constraints has not yet been solved efficiently. Similarly, the problem of moving range queries, where a scientist issues several range queries along a trajectory has not yet been studied.

The neuroscientists, for example, need to execute range queries interactively in close succession on a spatial model to analyze and visualize the surroundings of neuron elements along a neuron branch. The state-of-the art is to use a spatial index (e.g., the R-Tree [10]) and execute range queries on it repeatedly. Executing range queries repeatedly, however, is inefficient because every time a range query is executed, the overlap in the tree-like index structure introduces substantial overhead.

Research has so far only developed approaches which calculate safe regions for moving k nearest neighbor queries ([13] and others). If the query point moves into these regions, then the result does not need to be updated, thereby avoiding reading data from disk unnecessarily. The approaches developed for moving objects, however, do not support moving queries. New indexing approaches thus have to be developed that improve the performance of moving range queries, by making the execution of spatially close queries more efficient and by prefetching data.

Similarly, in advanced scenarios k nearest neighbor queries only need to retrieve the spatially closest elements that satisfy additional criteria. Neuroscientists will for instance use this type of query to retrieve the nearest postsynaptic neuron branches to a particular neuron. State of the art approaches do not execute this query efficiently [19, 20]. They either find a candidate set of nearest neighbors with a traditional kNN approach and then eliminate the candidates that do not satisfy the additional constraints, or filter all elements with the additional constraint and then find in the remainder the k nearest neighbors. None of these approaches, however, consider the pruning power of the additional constraints; and in the worst case all elements need to be visited. New kNN approaches therefore must make use of the pruning power of the additional constraint as early as possible in the query execution.

Many more indexes need to be developed to answer complex questions efficiently. Adding many potential candidate indexes of course also further complicates the problem of automated physical design.

4 Conclusions

In this paper, we identify the data management problem faced by researchers in two particular scientific disciplines, and identify the research challenges that need to be tackled in order to help scientists to manage their data automatically.

Because of the way scientists work with data, we identify the problems and the research challenges of continuous automated physical design and in-situ querying of raw data files. Research in the former is needed because scientists constantly revise their hypothesis, and hence data collected and queries asked change constantly. As a consequence, the physical design of the data has to be updated perpetually, challenging today's automated physical design methods. The latter is necessary because scientists often do not query all data collected. Importing

all data into a database hence is inefficient and there is a need for new in-situ querying approaches for raw data files.

We furthermore identify the problems of ever more fine grained models and complex queries in the simulation sciences. Both trends challenge current indexing approaches and open new opportunities for research. Developing new indexes at the same time complicates the physical design problem, challenging the state-of-the-art approaches in this area as well.

None of these challenges is trivial and will require considerable research. Nevertheless, interesting data management research problems can be tackled when helping scientists to automate their data management. Doing so will ultimately help scientists to focus on their research again.

References

[1] S. Agrawal et al. Database Tuning Advisor for Microsoft SQL Server 2005. In VLDB, 2004.

[2] H. Markram. The Blue Brain Project. *Nature Reviews Neuroscience*, 7(2):153–160, 2006.

[3] J. Gray, D. T. Liu, M. Nieto-Santisteban, A. S. Szalay, D. J. DeWitt, G. Heber. Scientific Data Management in the Coming Decade. In SIGMOD, 2005.

[4] ATLAS Experiment. http://atlas.ch/.

[5] D. Malon, E. May. Critical Database Technologies for High Energy Physics. In VLDB, 1997.

[6] I. Bird. Computing for the Large Hadron Collider. In Annual Review of Nuclear and Particle Science, 2011.

[7] POOL - Persistency Framework. http://pool.cern.ch/.

[8] CERN Advanced STORage manager (CASTOR). http://castor.web.cern.ch/.

[9] T. Brinkhoff, H. Kriegel, and B. Seeger. Efficient Processing of Spatial Joins R-Trees. *SIGMOD '93*.

[10] A. Guttman. R-trees: a Dynamic Index Structure for Spatial Searching. *SIGMOD '84*.

[11] V. Gaede and O. Guenther. Multidimensional Access Methods. *ACM Computing Surveys*, 30(2), 1998.

[12] T. K. Sellis, N. Roussopoulos, and C. Faloutsos. The R+-Tree: A Dynamic Index for Multi-Dimensional Objects. *VLDB '87*.

[13] X. Yu, K. Pu, and N. Koudas. Monitoring k-Nearest Neighbor Queries over Moving Objects. *ICDE '05*.

[14] C. Curino, E. P. C. Jones, R. A. Popa, N. Malviya, E. Wu, S. Madden, H. Balakrishnan, N. Zeldovich. Relational Cloud: A Database-as-a-Service for the Cloud. In CIDR, 2011.

[15] D. J. DeWitt and J. Gray. Parallel Database Systems: The future of High Performance Database Processing. In Magazine Communications of the ACM, 1992.

[16] D. C. Zilio, J. Rao, S. Lightstone, G. Lohman, A. Storm, C. Garcia-Arellanom and S. Fadden. DB2 Design Advisor: Integrated Automatic Physical Database Design. In VLDB, 2004.

[17] S. Idreos, I. Alagiannis, R. Johnson, A. Ailamaki. Here are my Data Files. Here are my Queries. Where are my Results?. CIDR 2011

[18] S. Idreos, M. Kersten, S. Manegold. Database Cracking. CIDR 2007

[19] G. Cong, C. S. Jensen, and D. Wu. Efficient Retrieval of the top-k Most Relevant Spatial Web Objects. *VLDB '09*.

[20] R. Hariharan, B. Hore, et al. Processing Spatial-Keyword (SK) Queries in Geographic Information Retrieval (GIR) Systems. *SSDBM '07*.

[21] D. Borthakur. The Hadoop Distributed File System: Architecture and Design. http://hadoop.apache.org/