SQL vs NoSQL for simulation data management

- Under development
- Approximate Bayesian Computation
- extremely scalable distributed databases
- simulation results

Abstract:

The ability to manage large amounts of simulation data is pivotal for exploring complex computational models in biology and elsewhere. One of the key debates of the last few months is whether a relational database and support for SQL queries is the "one size fits all" solution for emerging distributed processing paradigms. While expensive distributed SQL databases can excel at filtering data if properly optimised, current processing needs call for extreme scalability at low cost and uniform processing (i.e., a specific type of processing is applied all data, as opposed to selective subsets); this tendency has been referred to as the "NoSQL" movement.

In this project, our aim is to evaluate various options that arise in building extremely scalable cost-effective and easy-to-manage distributed databases of simulation results that support (i) automated replication and fault-tolerance to allow running on cheap (and often unreliable) hardware (ii) retrieval and annotation of individual simulation results, (iii) plotting one parameter against another for large numbers of arbitrarily selected single-run-results, and (iv) computation of summary statistics and other large-scale analytics for the stochastic output from repeated simulations of the same input parameter combination.

Commercial relational databases provide distributed storage support, but are (i) tailored towards a different processing paradigm (i.e., filter-based processing), and are (ii) not cost-effective (i.e., an annual license for a distributed version of a commercial product might far exceed other costs of ownership for the data).

Our goal is to implement a prototype system that supports the features above in an easy-to-use way. The target user of the system will be a research scientist interested in storing and uniformly manipulating the outcome of simulation results – a processing paradigm that manifests in multiple disciplines, including biology. To test the effectiveness of our general solution, we will apply our system to live production data from the evolution@home global computing system (>1 million single simulation runs worth >500 CPU years; a general system for exploring evolutionary models that is currently used to explore potential genetic causes for extinctions of endangered species).

In Detail:

Many complex problems in science today are addressed by computer simulations that map a set of input parameters with the help of a model to a set of output parameters. This pattern appears in multiple disciplines: biology, geology, physics, astronomy, to name but a few. Investigating the model is easy as long as the number of parameter combinations is relatively small. However, as soon as simulation projects grow beyond a certain size, the effectiveness of analyses is hampered by the need to search large amounts of often poorly organised data. A frequently suggested solution is to "simply" store such data in a relational database. While this helps structuring the data, relational databases are not optimised for the fast processing of scientific data and the size of such databases is mostly limited to a single machine. Furthermore, relational databases do not come with code that supports the most common use-cases of scientists; rather, the extra functionality needed for the analysis of the simulation results is implemented externally. The result is that the relational database acts as an elaborate storage layer. In fact, most scientific databases do not use a relational database back-end; rather, they operate over flat files in a file system, since most processing takes place in bulk.

From a structured data management perspective (i.e., assuming that flat files are not used and some semantics are readily supported by the storage layer) there are three fundamentally different approaches for achieving extreme scalability, which we use here to mean the virtually unlimited growth of nodes in a massively distributed shared-nothing database. Each approach has unique strengths and weaknesses:

1) Horizontal partitioning of tables in relational databases. Each node stores different rows of a huge table and handles all requests relating to these records. Advantages include fast access to all the values of a particular record (all on same machine), while access of particular columns across many records can be slower (on many different machines).
2) Vertical partitioning of tables in relational databases. Each node stores a different column (or part of it), resulting in faster access for queries that are frequently used by scientists (e.g., plot parameter x versus y). If access to all values of a particular record is rarely needed, slower speed for such requests will not be critical.
3) Distributed file systems for cluster-like processing. Files are organised in terms of records containing name-value pairs (‘columns’). The file system provides bindings for content-based retrieval, while a higher-level abstraction provides a way to process the files in bulk. MapReduce has emerged as a highly parallelisable data-flow for processing data in such an environment. Such systems allow for much larger flexibility as each file could carry different content; at the same time it becomes much harder to ensure the application-level correctness of data, as no schema is being enforced (as is the case in relational databases).

Stonebraker et al. (2010) have argued that all three approaches can, in principle, deliver the same functionality and extreme scalability. However, a careful comparison of all three approaches indicates different trade-offs that are required to get the corresponding systems to work (Pavlo et al. 2009; Stonebraker et al. 2010). For a scientist who simply wants to manage an extremely large dataset, but has no particular interest in relational databases the following practical details stand out.

Horizontal and vertical partitioning are not supported by mature open source systems, but only by big commercial relational databases. These (i) have a high licensing fee for large clusters, as each node requires a separate licence, (ii) due to their complexity require almost a full-time database administrator with specialist knowledge of both the database system and the data that is being stored, (iii) have a filtering performance that is potentially better than that of MapReduce systems, particularly when it comes to identifying data, but require specialist insights and sometimes even vendor help to optimise performance, as response time can increase dramatically if some tuning parameters have the wrong value. They can (iv) mimic cluster-processing functionality by implementing User-Defined-Functions (UDFs), but some scientists find these together with all required SQL code to be more difficult to implement than a custom-built system.

In comparison, cluster management systems offering a fully replicated storage layer and a MapReduce-like data processing paradigm are much simpler, both at a conceptual level and in their practical administration. They usually provide good out-of-the-box performance (albeit improving on that can be challenging) and many are freely available. They are particularly well suited for processing massive amounts of data through automatic data-flow parallelisation by using two functions: the “map” function is applied to identify and/or re-group the input into disjoints sets, and the “reduce” function to process each disjoint set in isolation and combine all individual results into the overall result. Such functionality is almost tailor-made for simulation-based studies. It is particularly well-suited to compute "multi-run-analytics" like the mean and variance for many stochastic single simulation runs (where a single set of input parameters results in many sets of output parameters combinations that capture the stochastic variations). Here the "map" phase groups results according to input parameter combinations, while the "reduce" phase computes the final statistics for each group. The relative ease of use associated with these systems and their out-of-the-box redundancy earned them widespread use.

Building on recent NoSQL approaches we will use best practices to develop a general scheme for organising simulation data in order to explore potential bottlenecks. Bad performance from poor data organisation is rarely compensated by efficient DBs. Thus the main aim to provide a general easy-to-use scheme for organising simulation data in an extremely scalable way. A core idea is that most queries of simulation data focus on one or a few simulation "projects". We will use the following principles to optimise our scheme for this use-case:
1. Keep projects separate so that searching in one does not require looking at any data in all others.
2. Distribute each project among as many data storage nodes as reasonable in order to increase parallelism and hence speed.
3. Group results for efficiency (eg. 64MB; reduce access time, fragmentation).
4. Keep stochastic repeats of the same input parameter combination on the same data storage node for fast "multi-run-analytics".
5. Avoid all indices to save insertion time; exception: important administrational indices like a hash on unique input parameter combinations.

We will evaluate various systems for cluster-based data management without a huge price-tag that a scientist in need of an extremely scalable data management solution might consider choosing. Our evaluation will be based on empirical comparisons of corresponding designs in order to evaluate (i) the potential limits of the scalability, (ii) fault-tolerance, (iii) ease of administration, (iv) ease of implementing special types of processing and (v) expected scalability of performance.

We plan to follow a two-phase process. First, we will evaluate the above factors in the two arguably leading such solutions: Hadoop, and CouchDB. If time permits and a fully functional version becomes available during the evaluation phase of the project, a potential third system is SciDB. Our evaluation will result in a report of the comparative study of these systems, which we plan to make freely available over the web to help other scientists find their system of choice.

We will then focus on an in-depth evaluation of the actual performance of the most promising system. To do this, we will design a general scheme for handling arbitrary simulation results by this system, including a facility for uploading and identifying simulation results by filtering based on input parameters. We will then implement various data processing algorithms for large-scale data analytics on top of the system. We will do so for the general case (e.g., averaging the output parameters of stochastic simulations over corresponding sets of input parameters, supporting queries needed for parameter estimation via Approximate Bayesian Computation as described in Beaumont & Rannala, 2004).

As a particular test-case of our proposed solution, and given the expertise of the participants in the project, we will import live production data from the evolution@home global computing system (>1 million single simulation runs worth >500 CPU years; a general system for exploring evolutionary models that is currently used to explore potential genetic causes for extinctions of endangered species). We will run the same general purpose processing pipelines developed above in order to test their performance in a real-world production setting and write a report about our experiences.

Our goal is to implement a system that supports the features above in an easy-to-use way in order to reduce the pain associated with data management for many scientists with a particular view to biologists who want to minimise the time required for obtaining workable solutions. In the end, we will have produced a prototype distributed data management system for simulation-based studies, which also provides the functionality for large-scale data analytics. In the process, the system will help address the question of whether a "NoSQL" data management system is capable of serving the needs of scientists.

References:
(1) Stonebraker et al. MapReduce and parallel DBMSs: friends or foes? Communications of the ACM (2010) vol. 53 (1)
(2) Pavlo et al. A comparison of approaches to large-scale data analysis. SIGMOD '09: Proceedings of the 35th SIGMOD international conference on Management of data (2009)
(3) Dean J & Ghemawat, S (2010) MapReduce: A Flexible Data Processing Tool Communications of the ACM (2010) vol. 53 (1)
(4) Cudre-Mauroux et al. A demonstration of SciDB: a science-oriented DBMS. Proceedings of the VLDB Endowment (2009) vol. 2 (2)
(5) Beaumont, M. A. & Rannala, B. 2004 The Bayesian revolution in genetics. Nat. Rev. Genet. 5, 251-261.

Who am I?:
Dr Laurence Loewe, currently postdoc in the Centre for Systems Biology at Edinburgh, formerly Lecturer in Evolutionary Genetics, School of Biological Sciences, University of Edinburgh; founder and chief developer of the evolution@home global computing system. Dr Stratis Viglas, Senior lecturer in Database Systems at the School of Informatics of the University of Edinburgh; substantial expertise in database query processing and optimisation and distributed systems.
How is it novel? What is exciting about it?:
At the moment scientists with a need for extremely scalable databases are faced with a bewildering array of systems that might potentially be used. In order to encourage more biologists to engage in data-intensive explorations of computational models, we will narrow down the options and implement a reasonable number of use-cases for our top candidate system. The prospect to have an extremely scalable system that supports most (or all) a biologist needs without the added price and management complexity of commercial relational databases is exciting. In addition, large-scale data analytics of simulation-based results is pivotal for investigating the reliability of many simulation models in science. Recent years have also seen increasing interest in Approximate Bayesian Computation (ABC), which facilitates likelihood-free inference for complex models that can be simulated, but whose likelihood function is not computable. ABC strongly depends on evaluating very large numbers of simulation results. Developing an extremely scalable database that supports ABC places this project on the cutting edge of both, ABC and scientific data management research.
What will I do next? What opportunities will it open up?:
The proof of principle developed here could serve as the basis for applying for a grant that expands this approach to make it more general, more user-friendly and more efficient in handling scientific data. For example, it would be desirable to find efficient algorithms for integrating the multidimensional access methods provided by HDF5 by including support for this very efficient low-level scientific file format. From strictly a database perspective, the prototype system will identify the bottlenecks associated with a distributed data management system that does not support a relational querying interface. As such, it can act as a starting step for a grant proposal that will aim to extend the functionality of such systems for relational database-like query processing, but without compromising their semi-structured nature. From a computing biologist's perspective, this work could support the data management side of grant applications that use massively distributed simulations for exploring complex models in areas like systems biology or evolution. The rising use of Approximate Bayesian Computation approaches will increase the need for such data management solutions.

What constitutes success? How risky is it?:
Success for this project will mean: 1) Deliver a report that compares various candidates for extremely scalable distributed scientific data management. 2) Implement the described stack of functionality for the most promising candidate. 3) Develop a general scheme for handling simulation results. 4) Measure how much additional distributed processing nodes will contribute to improving performance for large-scale analytics of simulation-based studies, using evolution@home results as a real-world test case. The risk of not completing these points is moderately low. The biggest challenge is in the aim to develop a “general scheme”. It will certainly work for a number of general problems that are shared between all scientific simulation studies, but being truly general demands being able to deal with all possible cases. While we will aim to get as much ground covered as possible, we think that achieving “generality” in this sense is a high-risk target.
What resources do I bring to the project?:
Dr Laurence Loewe from the School of Biological Sciences brings considerable expertise in analyzing biological simulation datasets and as chief developer of the evolution@home system is ideally positioned for exploiting this real world application for the purposes described here. He will do most of the work of the project and will bring about a million simulation results worth more than 500 CPU years from evolution@home to the project in order to provide a realistic test case. Dr Stratis Viglas is member of the database group of the School of Informatics and has considerable expertise in database design in general and query optimization and processing in particular. His expertise and interests in P2P computing and MapReduce are an added bonus to the project. He will contribute his expertise in discussions and advise on how to design various aspects of the system. Dr Jano van Hemert and his group support this project as discussion partners. We plan to discuss various aspects of our system design with him and other members of his group as appropriate. We anticipate that his expertise will provide valuable input for our project. Prof. Dr. Malcolm Atkinson has agreed to be a discussion partner for this project. We will discuss key decisions with him to benefit from his considerable experience with database, scalability and Grid issues.
What resources and expertise do I need?:
We request 4 person months by a software developer with considerable experience in evaluating DB systems and evolution@home simulation data to efficiently implement the system described here. Dr Laurence Loewe has explored various options with his current employer, the Centre for Systems Biology at Edinburgh. CSBE has agreed to defer his current contract for up to 6 months should this idea-lab project be funded. As lead developer and lead analysing biologist of evolution@home, he is ideally positioned to explore issues surrounding efficient ways of analyzing this large dataset. He has considerable experience in exploring potential designs for a system that can manage the evolution@home data and has recently been increasingly interested in ABC. The timing of this project is flexible, so a start after August 2010 is not a problem. We anticipate that results from this project scale approximately linearly with the time spent on it; thus shorter funding will be much better than no funding at all. We need two clusters with at least 3 computers in each, better more. One cluster will host the open system that shall serve as a test ground for third-party scientists to explore whether our system might work for them. The other cluster will be dedicated to hosting evolution@home results while analyzing the performance and scalability of that cluster (no third-party activity to get unbiased performance measures). We anticipate using a collection of unused old computers from the School of Informatics and elsewhere to do this (some machines have already been identified).
What shared resources, if any, will the project create?:
a) Documented experience. We will document in appropriate reports (i) the initial evaluation of extremely scalable system candidates, (ii) the in depth performance and scalability evaluation of the top candidate, some of it based on the live-production data accumulated by evolution@home, (iii) our design of an extremely scalable simulation results database for managing simulation data and (iv) practical how-to's that help novices to perform typical tasks. All these will be published online. b) An

open test installation. We will set up a test installation building on currently unused hardware in the School of Informatics. This installation can be used by third-party scientists, who simply want to try out the system we have developed or who want to use it for more serious data-analysis jobs (people with large demands will need to add servers and storage space).
What is the timescale?:
Months 1: We will do a survey of extremely scalable distributed data management systems and pick the best two for more in-depth analysis and practical tests. At the moment the top candidates are CouchDB and Hadoop. At the end of the month we will come up with a winner that we will use for the rest of the project. Months 2: We will develop a general design that will accept a wide range of simulation datasets from the community. Months 3-4: We will implement the open test cluster that will accept any simulation datasets from the community. We will advertise its existence so we can get some user feedback and improve things where possible while the project is still going on. At the same time we will set up the second cluster and move the evolution@home results to it in order to do some in-depth-testing of performance and scalability of various aspects under realistic conditions.