Benchmarking OODBs with a Generic Tool

Jérôme Darmont and Michel Schneider
Université Blaise Pascal – Clermont-Ferrand II, France

We present in this paper a generic object-oriented benchmark (OCB: the Object Clustering Benchmark) that has been designed to evaluate the performances of Object-Oriented Databases (OODBs), and more specifically the performances of clustering policies within OODBs. OCB is generic because its sample database may be customized to fit any of the databases introduced by the main existing benchmarks, e.g., OO1 (Object Operation 1) or OO7. The first version of OCB was purposely clustering-oriented due to a clustering-oriented workload, but OCB has been thoroughly extended to be able to suit other purposes. Eventually, OCB’s code is compact and easily portable. OCB has been validated through two implementations: one within the O2 OODB and another one within the Texas persistent object store. The performances of a specific clustering policy called DSTC (Dynamic, Statistical, Tunable Clustering) have also been evaluated with OCB.

The need to evaluate the performances of Object-Oriented Database Management Systems (OODBMSs) is important both to designers and users. Performance evaluation is useful to designers to determine elements of architecture, choose between caching strategies, and select Object Identifier (OID) type, among others. It helps them validate or refute hypotheses regarding the actual behavior of an OODBMS. Thus, performance evaluation is an essential component in the development process of efficient and well-designed object stores. Users may also employ performance evaluation, either to compare the efficiency of different technologies before selecting an OODBMS or to tune a system.

The work presented in this paper was initially motivated by the evaluation of object clustering techniques. The benefits induced by such techniques on global performances are widely acknowledged and numerous clustering strategies have been proposed. As far as we know, there is no generic approach allowing for their comparison. This problem is interesting for both designers (to set up the corresponding functionalities in the system kernel) and users (for performance tuning).

There are different approaches used to evaluate the performances of a given system: experimentation, simulation, and mathematical analysis. This paper focuses mainly on the first two approaches. Mathematical analysis is not considered because it invariably uses strong simplification hypotheses (Benzaken, 1990; Gardarin et al., 1995) and its results may well differ from reality.

Experimentation on the real system is the most natural approach and a priori the simplest to complete. However, the studied system must have been acquired, installed, and have a real database implanted in it. This database must also be significant of future exploitation of the system. Total investment and exploitation costs may be quite high, which can be drawbacks when selecting a product.

Simulation is casually used in substitution or as a complement to experimentation. It does not necessitate installing nor acquiring the real system. It can even be performed on a system still in development (a priori evaluation). The execution of a simulation program is generally much faster than experimentation. Investment and exploitation costs are very low. However, this approach necessitates the design of a functioning model for the studied system. The reliability of results directly depends on the quality and the validity of this model. Thus, the main difficulty is to elaborate and validate the model. A modelling methodology can help and secure these tasks.

Experimentation and simulation both necessitate a workload model (database and operations to run on this database) and a set of performance metrics. These elements are traditionally provided by benchmarks. Though interest for benchmarks is well recognized for experimentation, simulation approaches usually use workloads that are dedicated to a given study, rather than workloads suited to performance comparisons. We believe that benchmarking techniques can also be useful in simulation. Benchmarking can help validate a simulation model, as compared to experimental results or support—a mixed approach in which some performance criteria necessitating precision are measured by experimentation and other criteria that do not necessitate precision are evalu-
ated by simulation.

There is no standard benchmark for OODBs, even if the more popular benchmarks, OO1, HyperModel, and OO7 are de facto standards. These benchmarks are aimed at engineering applications (computer aided design, manufacturing, or software engineering). These general-purpose benchmarks feature quite simple databases and are not well suited to the study of clustering, which is very data-dependent. Many benchmarks have been developed to study particular domains. A fair number of them are more or less based on OO1, HyperModel, or OO7.

In order to evaluate the performances of clustering algorithms within OODBs, we designed our own benchmark: OCB (Darmont et al., 1998). It originally had a generic object base and was clustering-oriented through its workload. It actually appeared afterwards that OCB could become more general, provided the focused workload was extended, as described in this paper.

The objective of this paper is to present full specifications for a new version of OCB. More precisely, we address the following points: the generalization of the OCB workload so that the benchmark becomes fully generic, a comparison of OCB to the main existing benchmarks, and a full set of experiments performed to definitely validate OCB. These performance tests were performed on the OODB (Deux, 1991), the Texas persistent object store (Singhal et al., 1992), and the DSTC clustering technique (Bullat & Schneider, 1996). The results obtained are discussed in this paper.

We are aware of the legal difficulties pointed out by Carey et al. (1993, 1994). Indeed, OODBMS vendors are sometimes reluctant to see benchmark results published. The objective of our effort is rather to help software designers or users evaluate the adequacy of their product in a particular environment. OCB should also prove useful for researchers, to benchmark OODB prototypes and/or evaluate implementation techniques.

The remainder of this paper is organized as follows. The de facto standards in object-oriented benchmarking are briefly presented (OO1, HyperModel, and OO7; as well as the Justitia benchmark, which is interesting due to its multi-user approach). Next, our proposed benchmark, OCB, is described and compared to the other benchmarks. Experiments performed to validate our benchmark are also presented. We conclude the paper with future research directions.

RELATED WORK

The OO1 Benchmark

OO1, also referred to as the “Cattell Benchmark” (Cattell, 1991), was developed early in the 1990s when there was no appropriate benchmark for engineering applications. OO1 is a simple benchmark that is very easy to implement. It was used to test a broad range of systems, including object-oriented DBMS, relational DBMS, and other systems such as Sun’s INDEX (B-tree based) system. The visibility and simplicity of OO1 provide a standard for OODB benchmarking. A major drawback of this tool is that its data model is too elementary to measure the elaborate traversals that are common in many types of object-oriented applications, including engineering applications. Furthermore, OO1 only supports simple navigational and update tasks and has a limited notion of complex objects (only one composite hierarchy).

The HyperModel Benchmark

The HyperModel Benchmark (Anderson et al., 1990), also referred to as the Tektronix Benchmark, is recognized for the richness of the tests it proposes. HyperModel possesses both a richer schema and a wider extent of operations than OO1. This renders it potentially more effective than OO1 in measuring the performances of engineering databases. However, this added complexity also makes HyperModel harder to implement, especially since its specifications are not as complete as OO1’s. The presence of complex objects in the HyperModel Benchmark is limited to a composition hierarchy and two inheritance links. The scalability of HyperModel is also not clearly expressed in the literature, whereas other benchmarks explicitly support different and well-identified database sizes.

The OO7 Benchmark

OO7 (Carey et al., 1993) is a more recent benchmark than OO1 and HyperModel. It reuses their structures to propose a more complete benchmark and to simulate various transactions running on a diversified database. It has also been designed to be more generic than its predecessors and to correct their weaknesses in terms of object complexity and associative accesses. This is achieved with a rich schema and a comprehensive set of operations.

However, if OO7 is a good benchmark for engineering applications, it is not the case for other types of applications such as financial, telecommunication, and multimedia applications (Tiwary et al., 1995). Since its schema is static, it cannot be adapted to other purposes. Eventually, the database structure and operations of OO7 are nontrivial. Hence, the benchmark is quite difficult to understand, adapt, or even implement. Yet, to be fair, OO7 implementations are available by anonymous FTP.

The Justitia Benchmark

Justitia (Schreiber, 1994) has been designed to address the shortcomings of existing benchmarks regarding multi-user functionality, which is important in evaluating client-server environments. Justitia is also aimed at testing OODB capacity in reorganizing its database.

Because Justitia’s published specifications lack precision, the author’s work cannot be easily reused. Furthermore, taking multiple users into account renders the benchmark