The Features of BigOWLIM that Enabled the BBC's World Cup Website

Atanas Kiryakov, Barry Bishop, Damyan Ognyanoff, Ivan Peikov, Zdravko Tashev, Ruslan Velkov

Ontotext AD, 135 Tsarigradsko Chaussee, Sofia 1784, Bulgaria

ABSTRACT

Semantic repositories – RDF databases with inferencer and query answering engine – are set to become a cornerstone of the Semantic Web (and Linked Open Data) due to their ability to store and reason with the massive quantities of data involved. In this paper, we describe the features of BigOWLIM that have allowed it to penetrate into the commercial sector, focusing on one particular use-case, that being its use in the BBC's World Cup website.

General Terms

Query answering, Semantic Web, Full-text Search

Keywords

Database, Triple Store, RDF, SPARQL, OWL

1. INTRODUCTION

There is no formal definition of the term 'semantic repository' so for the purposes of this article we use this term for Database Management Systems (DBMS) that can be used to store, query and manage data structured according to the Resource Description Framework (RDF) standards [5]. Compared to Relational DBMS, such systems use flexible ontological schemata where data is processed by an inference-engine according to a well-defined semantics.

While semantic repositories have been around for more than a decade, so far they have not managed to win the hearts of a sizeable fraction of software architects. We believe there are two major reasons for this: immature tools and inconsistent feature sets. The first is a natural child illness of each new technology – mature tools can only appear on top of large user communities, which are not present for young technology. More worrying are some misconceptions about their essential features, which are widely spread across many of the providers and users of semantic repositories. We will point out two of those:

- Misconception 1: Reasoning is not an important feature; materialization does not work, all the required inference can be handled efficiently during query evaluation;
- Misconception 2: Data-partitioning is an important feature; it is the way to deal with critical constraints of the technology, e.g. performance and scalability.

We show that these are "urban myths" by presenting the basic design decisions behind BigOWLIM – a semantic repository which delivers best overall performance according to multiple independent evaluations conducted recently [2][11][9]. We will focus on features that appeared to be critical for the successful realization of BBC's World Cup 2010 web site, which was qualified in [7] as "*the first large scale, mass media site to be using concept extraction, RDF and a Triple store to deliver content.*"

2. BigOWLIM

OWLIM is a family of semantic repositories that provide storage, inference and novel data-access features delivered in a scalable, resilient, industrial-strength platform. The flagman of the family, BigOWLIM combines the robustness and scalability of relational databases, the reasoning capabilities of inference engines, and the efficiency of column stores in handling sparse data and evolving schemata. BigOWLIM delivers this functionality as an engine whose performance and resilience allowed it serve in the core of the semantic web publishing stack running the BBC's World Cup web site [8]. Here BigOWLIM handles millions of queries per day in a mission critical production environment, where the data is updated hundreds of times per hour.

BigOWLIM is also optimized to integrate and reason with linked data – these capabilities are proven in a couple of linked data portals (http://FactForge.net and http://LinkedLifeData.com), which provide public access to billions of linked data statements integrated from tens of datasets.

3. INFERENCE CAPABILITIES

The inferencing strategy in OWLIM is one of total materialization (apart from an optimization for **`owl:sameAs`** that is not discussed in this paper) based on R-Entailment (as defined by ter Horst [10]) where Datalog [3] like rules with inequality constraints operate directly on a single ternary relation that represents all triples.

Total materialization involves computing all the entailed statements at load time. While this introduces additional reasoning cost when loading statements in to a repository, the desirable consequence is that query evaluation can proceed extremely quickly.

Several standard rule sets are included in all editions of OWLIM and these include (in more or less increasing levels of complexity): 'empty' (no inference), OWL-Horst [10], RDFS [1], owl-max (RDFS plus most of OWL-Lite) and OWL2-RL [6].

In addition to the standard semantics, user-defined rule-sets can be used. In this case the user provides the full pathname to a custom rule file that contains definitions of axiomatic triples, rules and consistency checks.

4. RETRACTING ASSERTIONS

As mentioned above, OWLIM materializes all inferred statements at load time and whenever new statements are added to the repository. This has the highly desirable advantage that query answering is very fast, due to the fact that no further inference needs to be done. Updates that simply add new statements are treated in the same way as at load time, i.e. new statements are fed to the inference engine that applies the inference rules (making joins across new statements with existing statements) until no new inferences are obtained. Since the semantics (both standard and custom) must be monotonic, insert operations incrementally add to the set of explicit and inferred statements. However, retracting explicit statements that are used to infer other statements becomes more complicated. In SwiftOWLIM, this is achieved by simply invalidating all inferred statements and re-computing the full-closure whenever an update is committed. This has the advantage of simplicity of implementation, but the disadvantage of poor update performance and lack of scalability.

BigOWLIM has a specific optimization for handling delete operations that updates the full-closure incrementally. This technique labels statements to be deleted and then uses forward-chaining to identify those statements that can be inferred from them, followed by backward chaining to identify those inferred statements that are still supported by other means.

The result is that delete performance is only slightly worse than the insertion of new statements. This allows the repository to handle rapidly changing data even when answering queries over tens of billions of statements.

5. TRANSACTION CONTROL

OWLIM supports the 'read committed' transaction isolation level. It guarantees that changes will not impact query evaluation, before the entire transaction they are part of is successfully committed. It does not guarantee that execution of a single transaction is performed against a single state of the data in the repository. Regarding concurrency, multiple update/modification/write transactions can be initiated and stay open simultaneously, i.e. one transaction does not need to be committed in order to allow another transaction to complete Furthermore, update transactions are processed in sequence and do not block read requests in any way, i.e. hundreds of SPARQL queries can be evaluated in parallel (the processing is properly multi-threaded) while update transactions are being handled on separate threads.

One should note that OWLIM performs materialization, making sure that all the statements which can be inferred from the current state of the repository are indexed and persisted. By the time the commit method completes, all reasoning activities related to changes introduced by the corresponding transaction will have already been performed.

6. REPLICATION CLUSTER

BigOWLIM can be used in a cluster configuration where replication is used to improve resilience and provide scalable query answering.

The query performance of the cluster represents the sum of the throughputs that can be handled by each of the instances. In a simple configuration of 3 or 4 worker nodes, hundreds of thousands of query requests can be answered per hour while at the same time processing thousands of updates per hour – with non-trivial inference.

In a cluster configuration, there are two types of nodes: Masters and Workers. Masters act as the gateway to the cluster and all read/write requests go through these nodes. A cluster can have more than one master node, but only one is allowed to operate in read/write mode. The other master nodes operate in read-only mode, otherwise known as 'hot-standby'. They can be used for marshalling read requests and can take over handling updates if the current read/write master fails. Worker nodes are standard BigOWLIM instances exposed by the Sesame HTTP server – a servlet running in Tomcat or similar. Read and write requests are passed to the workers from the master nodes. This simple arrangement allows for a great deal of flexibility in the design of a cluster topology. The example given in Fig. 1 has two master nodes and three worker nodes. At any moment in time, clients of the cluster can send read requests (queries) to either master node, but updates can only be handled by the master in read/write mode. If this master node should fail, the hot standby master can be brought in to read/write mode and from then on will handle both read requests and updates, as well as taking over responsibility for ensuring the synchronization of all the worker nodes.

Each master node implements a JMX MBean [4] that is accessible using standard Java instrumentation tools, such as JConsole, and can be used to monitor and control the cluster while it is running. Typical activities supported include the monitoring of the health of each node, statistics gathering, adding and removing worker nodes.

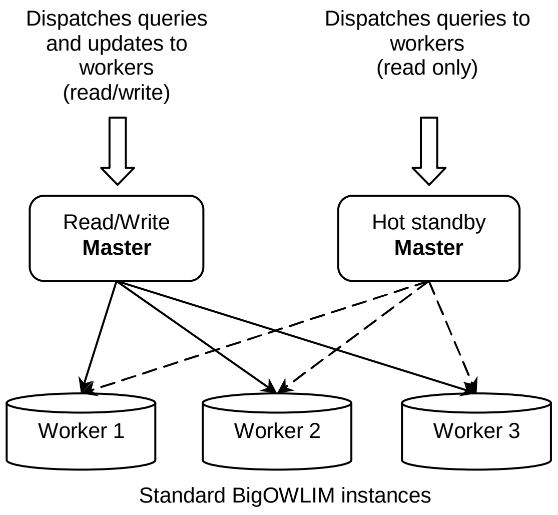
Dispatches queries and updates to workers (read/write)
Dispatches queries to workers (read only)
Read/Write Master
Hot standby Master
Worker 1
Worker 2
Worker 3
Standard BigOWLIM instances

Fig. 1 A typical replication cluster configuration

During normal operation, a master node will keep track of the size of each worker's read request queue, such that each read request is sent to the worker with the shortest read queue. Update requests are handled differently. First of all, the update is tested against a single worker node. If the update is successful and subsequent consistency checks pass then the update request is considered 'safe' and is passed to the rest of the worker nodes. Master nodes take additional care to ensure that the states of all worker nodes are properly synchronized and if an anomaly is detected, the problem worker node is released from the cluster. The monitor and control JMX interface can be used to return worker nodes to the cluster and initiate their synchronization.

In the event of a failure of a worker node, the performance degradation is graceful with respect to the number of healthy workers. The cluster can remain operational with just a single worker node.

7. CONCLUSION

The emerging Web of data has provided new challenges for software components that must expose this data and enable its widespread consumption. The OWLIM family of semantic repositories is ideally suited to this task due to its ability to store, reason and answer queries using the massive datasets involved.

OWLIM's development over the last 6 years was driven by pragmatic design decisions aimed to meet the requirements of a range of real-world applications, using it for data integration, metadata management and multi-paradigm information retrieval techniques that combine structured queries and reasoning on the one hand with full-text search and co-occurrence analysis on the other. This allowed OWLIM to develop to the point of maturity and comprehension which allowed it to serve as the back end for such a high-profile application as the BBC's World Cup 2010 web site. This use case demonstrated the viability of several design decisions:

- Distributed configuration, based on data replication, is ideal for applications where resilience and horizontal scalability with respect to query loads are key; in such environments data partitioning is inefficient and inappropriate;
- Reasoning based on forward chaining and materialization provides very good overall performance. When paired with intelligent

retraction techniques, it can cope with large numbers of updates, while simultaneously dealing with heavy query loads.

OWLIM continues to evolve with various new features planned for the near future. The next release of OWLIM will include enhanced support for geo-spatial data and some of the widely accepted geo-spatial vocabularies. Specialized indices will be used to access spatial data and a range of SPARQL extension functions will allow for expressive queries using 2D and 3D geometry.

The next release will also include interfaces that support the JENA RDF framework, enabling OWLIM to be used with both Sesame and JENA, the two most widely used RDF frameworks.

The current set of advanced features and world-leading performance have helped to position OWLIM as the semantic repository of choice for all environments that manage RDF data, particularly for Web-scale applications. The future evolution of OWLIM towards better compatibility and even more powerful data management features will ensure the continued uptake of this technology.

8. REFERENCES

1. Brickley, D.; Guha, R.V, *RDF Vocabulary Description Language 1.0: RDF Schema*, W3C (10 Feb 2004) http://www.w3.org/TR/rdf-schema
2. Bizer, Ch., Schultz, A.: *BSBM Results for Virtuoso, Jena TDB, BigOWLIM* (November 2009). http://www4.wiwiss.fu-berlin.de/bizer/Berlin SPARQLBenchmark/results/V5/index.html
3. Hervé Gallaire, Jack Minker (Eds.): *Logic and Data Bases*, Symposium on Logic and Data Bases, Centre d'études et de recherches de Toulouse, 1977. Advances in Data Base Theory, Plenum Press, New York, 1978.
4. *Java Management Extensions (JMX)*, homepage: http://download-llnw.oracle.com/javase/1.5.0/ docs/guide/jmx/
5. Klyne, G; Carrol , J. J; (eds). (2004). *Resource Description Framework (RDF): Concepts and Abstract Syntax*. W3C Recommendation 10 Feb. 2004. http://www.w3.org/TR/rdf-concepts/
6. Motik, B; Cuenca Grau, B; Horrocks, I; Wu, Z; Fokoue, A; Lutz, C. (eds.) (2009). *OWL 2 Web Ontology Language Profiles*. W3C Candidate Recommendation 11 June 2009. http://www.w3.org/TR/owl2-profiles/
7. O'Donovan, J. *The World Cup and a call to action around Linked Data*. BBC blog post. http://www.bbc.co.uk/blogs/bbcinternet/2010/07/the_world_cup_and_a_call_to_ac.html
8. Rayfield, J. *BBC World Cup 2010 dynamic semantic publishing"*, BBC blog post. http://www.bbc.co.uk/blogs/bbcinternet/2010/07/bbc_world_cup_2010_dynamic_sem.html
9. Stoilos G., Grau B. C., Horrocks I. *How Incomplete is your Semantic Web Reasoner?* In Proc. of the 20th Nat. Conf. on Artificial Intelligence (AAAI 10), 2010
10. ter Horst, H. J. *Combining RDF and Part of OWL with Rules: Semantics, Decidability, Complexity*. In Proc. of ISWC 2005, Galway, Ireland, November, 2005, pp. 668-684
11. Thakker, D., Osman, T., Gohil, S., Lakin, P, *A Pragmatic Approach to Semantic Repositories Benchmarking*. In Proc. of the 7th Extended Semantic Web Conference, ESWC 2010.