RDF Triple Stores for Client-side Applications

Joseph Utecht

July 2015

1 Introduction

Client driven web applications are becoming increasingly popular [8]. At the same time REpresentation State Transfer (REST) protocals for datasets have become more standardized, allowing for Javascript frameworks to implement standardized create, read, update and delete (CRUD) functions. This presents some interesting problems when working with current RDF triple stores [7].

2 Material and Methods

2.1 RDF Triple Stores

We have decided to focus on Apache Jena [1], Blazegraph [2], and Sesame [4] RDF stores for our testing due to their support for RDFS reasoning, open source licenses, ability to handle large datasets and REST endpoints for interaction. [10]

Testing was performed on VirtualBox VM [6] with 2x 2.4 GHz 6-core Xeon processors and 8GB of ram. The operating system was CentOS 7 [3] and all of the stores were run through the Java application server Tomcat [5].

Apache Jena uses a Java based front-end called Fuseki these are distributed under the Apache License 2.0. Fuseki can either be run standalone or under a Java application server. For the testing the most recent version Fuseki 2.0.0 under Tomcat was used with RDFS level reasoning.

Systrap's Blazegraph (formerly known as Bigdata) is distributed under either the GPLv2 or a commercial license. It can also either be run as a standalone or in a Java application server. Blazegraph 1.5.1 under Tomcat was used for testing with a triple + inference graph.

Much like Jena, Sesame is a framework that also includes a web front-end and native triple store. Sesame version 2.8.4 was used for testing with a native triple store with RDFS inference.

2.2 Testing Method

We selected the Lehigh University Benchmark (LUBM) [9] to test performance and capability of the triple stores. We did not use the default testing queries

with this dataset as they were more designed to measure the performance of OWL inference models, however this benchmark gave us an arbitrarily large set of data with RDFS classes that we could use for load testing.

To service a high performing client-side application the triple store must be able to both return queries quickly and in large volume. We built a testing framework to benchmark the triple stores we were testing. This framework measured three areas, data loading time, large query runtime, and rate small queries could be returned in parallel as queries per second.

Data load time was measured by breaking the LUBM dataset down into n3 formatted triples and then measuring both the time it took to load one million of those triples and the average load time per triple.

Large queries were defined as a query that would return greater than 5% of the total input size or more than 50,000 results in this instance. We used the following query to return this large dataset.

Small queries came in two varieties based on their complexity and was a query that would return less than 1% of the total input size, or 1,000 results.

3 Results

4 Discussion

References

[1] Apache jena. `https://jena.apache.org/`.

[2] Blazegraph. `http://www.blazegraph.com/bigdata`.

[3] Centos. `https://www.centos.org/`.

[4] Sesame. `http://rdf4j.org/`.

[5] Tomcat. `http://tomcat.apache.org/`.

[6] Virtualbox. `https://www.virtualbox.org/`.

[7] Robert Battle and Edward Benson. Bridging the semantic Web and Web 2.0 with Representational State Transfer (REST): Semantic Web and Web 2.0. *Web Semantics: Science, Services and Agents on the World Wide Web*, 6(1):61–69, 2008.

[8] R.T. Fielding and R.N. Taylor. Principled design of the modern Web architecture. *Proceedings of the 2000 International Conference on Software Engineering. ICSE 2000 the New Millennium*, 2(2):115–150, 2000.

[9] Yuanbo Guo, Zhengxiang Pan, and Jeff Heflin. LUBM: A benchmark for OWL knowledge base systems. *Web Semantics*, 3(2-3):158–182, 2005.

[10] Martin Voigt, Annett Mitschick, and Jonas Schulz. Yet another triple store benchmark? Practical experiences with real-world data. *CEUR Workshop Proceedings*, 912(Sda):85–94, 2012.