UNIVERSITÄT PADERBORN
Die Universität der Informationsgesellschaft
Faculty for Computer Science, Electrical Engineering and Mathematics
Department of Computer Science
Research Group DICE Group

Bachelor's Thesis

Submitted to the DICE Group Research Group
in Partial Fullfilment of the Requirements for the Degree of

Bachelor of Science

Basilisk – Continuous Benchmarking for Triplestores

by
Fabian Rensing

Thesis Supervisor:
Prof. Dr. Axel-Cyrille Ngonga Ngomo

Paderborn, January 7, 2022

Erklärung

Ich versichere, dass ich die Arbeit ohne fremde Hilfe und ohne Benutzung anderer als der angegebenen Quellen angefertigt habe und dass die Arbeit in gleicher oder ähnlicher Form noch keiner anderen Prüfungsbehörde vorgelegen hat und von dieser als Teil einer Prüfungsleistung angenommen worden ist. Alle Ausführungen, die wörtlich oder sinngemäß übernommen worden sind, sind als solche gekennzeichnet.

Ort, Datum

Unterschrift

Abstract. Abstract

Contents

1 Introduction

In the field of Semantic Web, knowledge graphs are an important structure to represent data and its relationships. To easily store and query the data in these knowledge graphs, some data structure or database is needed. The special kind of database developed to store knowledge graphs are called Triplestores.

Since knowledge graphs can contain huge amounts of data which can also be subject to many changes, Triplestores need to be able to handle many different workloads. Some scenarios need to handle huge amount of data being added, while others need to handle a lot of changes on the current data. To better test and compare Triplestores in these diverse scenarios, benchmarks are performed to allow an appropriate comparison between different Triplestores[11].

In general, Benchmarks are used to measure and compare the performance of computer programs and systems with a defined set of operations. Often they are designed to mimic and reproduce a particular type of workload to the system. In the context of Triplestores, a benchmark usually consists of creating a given knowledge graph on which multiple queries and operations are performed.

Often Triplestores are developed in long iterations and are bench-marked only in a late stage of such an development iteration. Today benchmarks and the evaluation of their results are usually done manually and bind developers time. Thus, performance regressions are found very late or never.

Several benchmarks for Triplestores have been proposed [11]. IGUANA is a benchmark-independent execution framework [5] that can measure the performance of Triplestores under several parallel query request. Currently the benchmark execution framework needs to be installed and benchmarks need to be started manually. Basilisk is a continuous benchmarking service for Triplestores which internally uses IGUANA to perform the benchmarks. The idea is that the Basilisk service will check automatically for new versions of Triplestores and start benchmarks with the IGUANA framework. Further it should be possible to start custom benchmarks on demand. If a new version is found in a provided GitHub- or DockerHub-repository, Basilisk will automatically setup a benchmark environment and starts a benchmarking suite.

This means that developers do not have to worry about performing benchmarks at different stages of development.

In this thesis we continue the development of the Basilisk platform and deploy an instance to a publicly available virtual machine.

The thesis is structured as follows. In Chapter 2 we take a look at the state of the art of Triplestore benchmarking. Chapter 3 introduces the fundamental concepts and topics to understand this thesis. The chapter 4 describes the architecture use in the Basilisk platform.

2 Related Work

This chapter reviews the state of the art of Triplestore benchmarking.

Several benchmarks have been proposed and developed. Many of these existing benchmarks each focus on different goals and scenarios to test the Triplestores.

2.1 Synthetic Benchmarks

The LUBM Benchmark[7] is a synthetic benchmark which focuses on the reasoning and inferencing capabilities of the Triplestores under test. The test data is about the university domain and can be generated to arbitrary size. The benchmark provides fourteen extensional queries that represent and test a variety of properties.

Another synthetic benchmark is SP^{2}Bench[12]. The data generated is placed in the DBLP scenario and it is tried that key characteristics and word distributions are close to the original DBLP dataset. The provided queries are mostly complex and the mean size of the result sets is above one million[10]. They also test for SPARQL features like union and optional graph patterns.

The WatDiv suite generates a synthetic benchmarks and consists of multiple tools[3]. First the data generator which generates scalable and customizable datasets based on the WatDiv data model schema. The query template generator generates diverse query templates which will then be used to generate actual queries. The queries get generate with the query generator which instantiates the templates with actual RDF terms from the generated dataset. For each template multiple queries can be generated. The benchmark only focuses on SELECT queries that does not make use of Union and Optional patterns

2.2 Benchmarks Using Real Data

FEASIBLE is a benchmark generation framework which generates datasets and queries from provide query logs[10]. This has the advantage that the data used for the benchmark could stem from queries about a special real world topic rather than an abstract synthetic model. FEASIBLE can also generate queries for the other SPARQL query types beside SELECT.

2.3 Benchmark Execution Frameworks

Hobbit framework ? needed?

The IGUANA framework is a benchmark independent benchmark framework. It is used in the Basilisk platform and is shortly described in section 3.1.4.

3 Background

This chapter explains the fundamental topics required to understand this thesis.

3.1 Semantic Web Topics

The following topics come from the research area of Semantic Web. Since this thesis focuses mostly on the implementation and deployment of the Basilisk framework, these topics are only introduced to give a basic understanding if the context in which the Basilisk framework is used.

3.1.1 Knowledge Graph

Knowledge Graphs are graphs intended to represent knowledge of the real world or smaller scenarios. The knowledge stored in Knowledge Graphs is modeled in a graph-based structure. Nodes represent entities which are connected by various types of relations, represented by labeled edges in the graph. This has the benefit to represent complex relations between different nodes and edges[9].

The simplest knowledge graph consists of three elements. The subject entity, the object entity and the labeled edge between them describing their relation. This atomic data entity is called triple.

In figure 3.1 a simple example of a knowledge graph is shown.

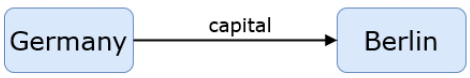

Figure 3.1: Simple Knowledge Graph

Since a graph structure is hard to store in a classic relational database a different type of storage is needed. The special kind of database developed to store knowledge graphs are called Triplestores.

3.1.2 Triplestore

Triplestores are a special kind of database developed to easily store and access knowledge graphs through queries. Example of Triplestores are Tentris[4], GraphDB[1], Virtuoso[2], or Jena TDB[3].

This thesis focuses on Triplestores that accept SPARQL queries, since the used benchmark framework IGUANA is using the SPARQL endpoint to perform benchmarks[5].

3.1.3 SPARQL

SPARQL (SPARQL Protocol and RDF Query Language)[8] is a query language for manipulating and retrieving data stored in Triplestores. Queries can contain optional graph patterns, conjunctions, disjunctions, as well as aggregation functions

3.1.4 IGUANA

IGUANA is a SPARQL benchmark execution framework[5]. The framework uses the SPARQL endpoint of the Triplestore under test to load, update and query the data. It allows the measurement of the performance during loading and updating of data as well as parallel requests to the Triplestore. IGUANA is independent of any benchmarks which allows it to run in different configurations and with different existing benchmarks and datasets.

3.2 Software Development

The following topic can be grouped under the field of software development.

3.2.1 Benchmark

Benchmarks for databases consist data set and sets of operations which will be performed on the data set. These operations are designed to simulate a particular type of workload to the system. The goal of a benchmark is to measure different metrics for a better comparison between various systems. Metrics used for databases and Triplestores are e. g., number of executed queries and queries per second[1].

A distinction is made between micro and macro benchmarks. Micro benchmarks focus on testing the performance of single components of a system. Macro benchmarks test the performance of a system as a whole. The benchmarks performed by the Basilisk platform set up in this thesis will only perform macro benchmarks.

3.2.2 Microservice

A microservice is an independently deployable piece of software that only implements functionalities that are closely related to the main task of the service [6]. The microservice interacts via messages through a defined protocol with other services.

3.2.3 Microservice Architecture

A microservice architecture is a way of designing a software application as a set of microservices which interact with each other to provide the designed functionality [6][2]. The functionality of the application gets split up into microservices which interact only through a defined protocol

[1] `https://graphdb.ontotext.com/`
[2] `https://virtuoso.openlinksw.com/`
[3] `https://jena.apache.org/documentation/tdb/`

of messages. This allows for a distributed system in which the individual service could be implemented in different programming languages. Also the microservices can be individually deployed and managed.

4 Approach

The basic architecture pattern of the Basilisk platform is the microservice architecture (see chapter 3.2.3 for a short description). This means that the platform is dividable into multiple parts which could run on different hardware systems.

There are three main services. These services communicate via message queues.

The Hooks Checking Service which regularly polls the Github or Dockerhub repositories for new versions of the observed Triplestores. The Jobs Managing Service processes the requests coming from the web-frontend, checks if the Hooks Checking Service has found a new version for a benchmark and creates jobs for new benchmarks. Lastly the Triplestore Benchmarking Service executes the benchmarks given to it and saves the results to a database.

The services will be deeper explained in the following sections.

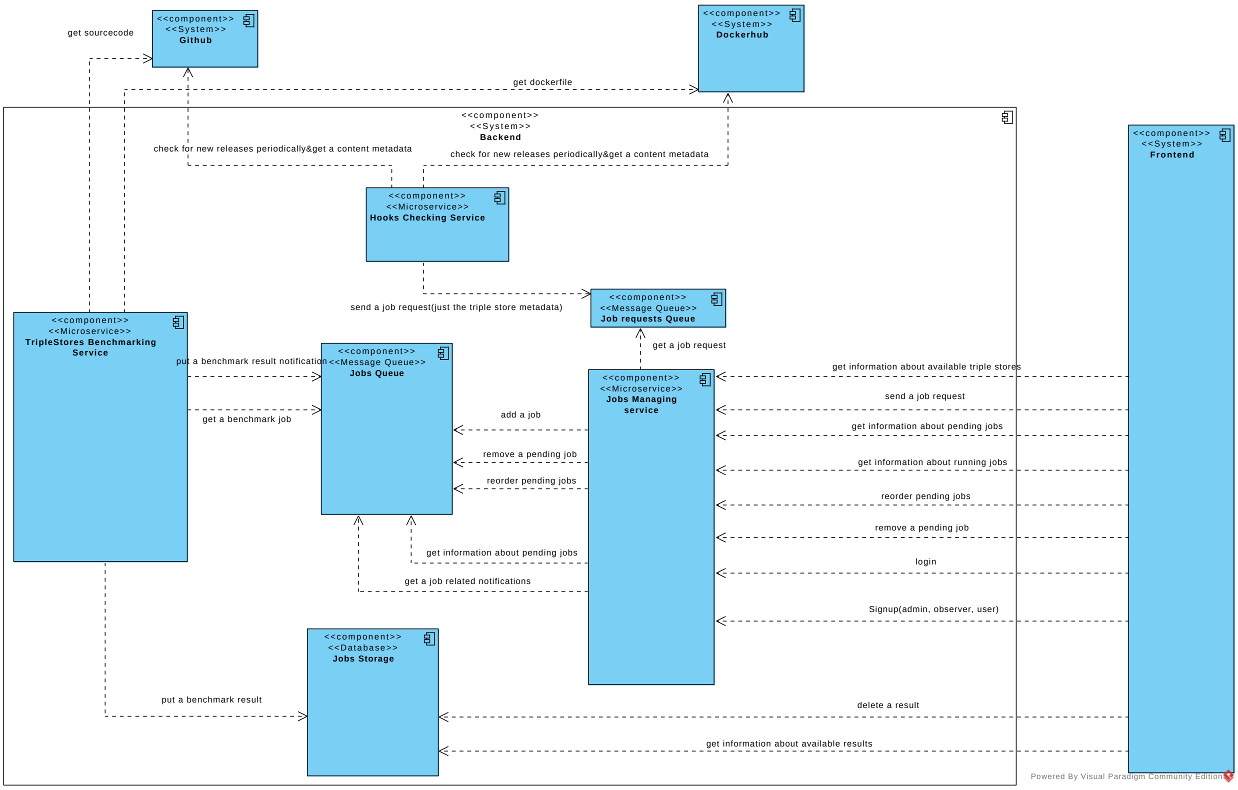

Figure 4.1: High level design of the Basilisk framework

5 Implementation

6 Evaluation

- Experiment setup, requirements - Performing of benchmarks - Result evaluation

7

Summary and Discussion

- Summary of the work - Highlighting the key findings of the evaluation stage

Bibliography

[1] Metrics - Iguana 3.3 Documentation, http://iguana-benchmark.eu/docs/3.3/usage/metrics/.

[2] Microservices, https://martinfowler.com/articles/microservices.html.

[3] Güneş Aluç, Olaf Hartig, M. Tamer Özsu, and Khuzaima Daudjee. Diversified Stress Testing of RDF Data Management Systems. In Peter Mika, Tania Tudorache, Abraham Bernstein, Chris Welty, Craig Knoblock, Denny Vrandečić, Paul Groth, Natasha Noy, Krzysztof Janowicz, and Carole Goble, editors, *The Semantic Web – ISWC 2014*, Lecture Notes in Computer Science, pages 197–212. Springer International Publishing.

[4] Alexander Bigerl, Felix Conrads, Charlotte Behning, Mohamed Ahmed Sherif, Muhammad Saleem, and Axel-Cyrille Ngonga Ngomo. Tentris – A Tensor-Based Triple Store. In Jeff Z. Pan, Valentina Tamma, Claudia d' Amato, Krzysztof Janowicz, Bo Fu, Axel Polleres, Oshani Seneviratne, and Lalana Kagal, editors, *The Semantic Web – ISWC 2020*, volume 12506 of *Lecture Notes in Computer Science*, pages 56–73. Springer International Publishing.

[5] Felix Conrads, Jens Lehmann, Muhammad Saleem, Mohamed Morsey, and Axel-Cyrille Ngonga Ngomo. Iguana: A Generic Framework for Benchmarking the Read-Write Performance of Triple Stores. In Claudia d' Amato, Miriam Fernandez, Valentina Tamma, Freddy Lecue, Philippe Cudré-Mauroux, Juan Sequeda, Christoph Lange, and Jeff Heflin, editors, *The Semantic Web – ISWC 2017*, volume 10588 of *Lecture Notes in Computer Science*, pages 48–65. Springer International Publishing.

[6] Nicola Dragoni, Saverio Giallorenzo, Alberto Lluch Lafuente, Manuel Mazzara, Fabrizio Montesi, Ruslan Mustafin, and Larisa Safina. Microservices: Yesterday, today, and tomorrow.

[7] Yuanbo Guo, Zhengxiang Pan, and Jeff Heflin. LUBM: A benchmark for OWL knowledge base systems. 3(2):158–182.

[8] Steve Harris, Andy Seaborne, and Eric Prud'hommeaux. SPARQL 1.1 Query Language, https://www.w3.org/TR/sparql11-query/.

[9] Aidan Hogan, Eva Blomqvist, Michael Cochez, Claudia D'amato, Gerard De Melo, Claudio Gutierrez, Sabrina Kirrane, José Emilio Labra Gayo, Roberto Navigli, Sebastian Neumaier, Axel-Cyrille Ngonga Ngomo, Axel Polleres, Sabbir M. Rashid, Anisa Rula, Lukas Schmelzeisen, Juan Sequeda, Steffen Staab, and Antoine Zimmermann. Knowledge Graphs. 54(4):71:1–71:37.

[10] Muhammad Saleem, Qaiser Mehmood, and Axel-Cyrille Ngonga Ngomo. FEASIBLE: A Feature-Based SPARQL Benchmark Generation Framework. In Marcelo Arenas, Oscar Corcho, Elena Simperl, Markus Strohmaier, Mathieu d' Aquin, Kavitha Srinivas, Paul Groth, Michel Dumontier, Jeff Heflin, Krishnaprasad Thirunarayan, Krishnaprasad Thirunarayan, and Steffen Staab, editors, *The Semantic Web - ISWC 2015*, Lecture Notes in Computer Science, pages 52–69. Springer International Publishing.

[11] Muhammad Saleem, Gábor Szárnyas, Felix Conrads, Syed Ahmad Chan Bukhari, Qaiser Mehmood, and Axel-Cyrille Ngonga Ngomo. How Representative Is a SPARQL Benchmark? An Analysis of RDF Triplestore Benchmarks. In *The World Wide Web Conference*, pages 1623–1633. ACM.

[12] Michael Schmidt, Thomas Hornung, Georg Lausen, and Christoph Pinkel. SP2Bench: A SPARQL Performance Benchmark.