Karlsruhe Institute of Technology

# Benchmarking of Graph Databases - Suitability for the Industrial Environment

Bachelor's Thesis
by

**Christian Navolskyi**

Chair of Pervasive Computing Systems/TECO
Institute of Telematics
Department of Informatics

| | |
|---|---|
| First Reviewer: | Prof. Dr. Michael Beigl |
| Second Reviewer: | M.Sc. Andrei Miclaus |
| Supervisor: | |

Project Period: 01/01/2018 – 30.04.2018

Ich versichere wahrheitsgemäß, die Arbeit selbstständig angefertigt, alle benutzten Hilfsmittel vollständig und genau angegeben und alles kenntlich gemacht zu haben, was aus Arbeiten anderer unverändert oder mit Abänderungen entnommen wurde.

Karlsruhe, den TODO: date

## Zusammenfassung

TODO: Zusammenfassung (Deutsch)

## Abstract

TODO: Zusammenfassung (Englisch)

# Contents

# 1. Introduction

## 1.1 Problem Statement

TODO: Highlight that current graph database benchmark papers are not covering our field. With the growing digitalisation of the industry more data is available and can be used to improve production processes. The amount of data created depends on the individual use case, but still it needs to be stored to be useful. Since there are multiple databases available it can be difficult to choose the right one for an individual scenario.

### 1.1.1 Use Case - Industry 4.0

There are multiple analytic algorithms to run on data to extract certain features. In the industry those algorithms play an important role too, but in this thesis we are looking at different aspects of the industrial use case, mainly inserting data and reading data. Though we have no direct partner in the industry for this thesis we try to TODO: No industry but trying our best to get good data / structures / workloads

#### 1.1.1.1 Inserting Data

To digitalise the production processes the data produced by every machine in the production line should be stored for future analysis. And to store that data it needs to be written into a database. Since most factories running 24 hours a day the machines are producing a lot of data during the day. That will be the base load for the underlying database, to store all that data from the production machines.

#### 1.1.1.2 Reading Data

Besides using the stored data for analysis algorithms, simply reading data from the database is another common use case. An example would be to get the time at which a specific product was processes by a specific machine to check if all parameters were set correctly.

## 1.2 Question

This thesis should give an answer to the question, if graph databases are suitable for an industrial application. Suitable in this case means that the database can withstand the amount of data written to it during production. In chapter 3 section 3.1 we analyse how much data could be written to a database and of which structure that data is. We motivate graphs and the use of graph databases in section 2.1 and section 2.3 respectively and will also concentrate on those in this thesis, because of the structure of data from production.

## 1.3 Methodology

We will chose the databases to use for our testing from other studies covering benchmarking graph databases to be able to compare the results and look to similarities in behaviour. To evaluate different databases we first will look up existing benchmarks and choose the best fitting one for our research. In the benchmarking program we need to look at the creation of data and how it can be stored and retrieved. The same exact dataset should be used for all databases equally to eliminate the variation that comes with generating data during each benchmark run. Next the workloads should be customisable and be able cover our goals. Also the measurements taken during the benchmark need to be useful for us. The databases should be able to connect to the benchmark in the same way.

## 1.4 Goal of this Thesis

With this thesis we want to examine whether and if so, how well graph databases are able to stand the load of an production line. Because every manufacturer is different and we cannot cover all scenarios we try to cover the most important parameters so that the suitability for the individual case can be estimated.

## 1.5 Structure

In chapter 2 we are motivating graph and the use of graph databases. The different kinds of graph databases are explained and an example database which we are testing is mentioned and shortly described. Also in this chapter we are comparing the different available benchmarking programs and their features. NOTE: Maybe mention related work

In chapter 3 the industrial data is modelled and its structure is analysed as well as a reasonable amount of data is determined. Then we are figuring out how a workload could look like in an industrial environment. At last we further analyse out chosen benchmarking program and give an overview of its procedure.

Chapter 4 is focused on the design of the different extensions for the benchmark and also the concrete data structure. For the extension we cover the design of the specific workloads, the design of classes to create and recreate the dataset, the graph workload class managing the graph databases and the graph data and finally the database bindings which are responsible for connecting the database to the benchmarking program.

In chapter 5 the implementation of the single components is described. First we cover the graph data generator which includes the class for creating the graph data as well as the class for recreating it from files. Next the bindings are implemented and their individual adaptions to the benchmark are highlighted. And lastly we explain the graph workload class which is the mediator between the created graph data and the database bindings.

Chapter 6 focuses on running the benchmark and evaluating the results. First, we define our objective during evaluation. Then the configuration of our system is stated, as well as the hardware as the software side. Next the procedure of running the benchmarks sequentially is explained following be the different aspects we are testing. These are grouped into ”maximum load” in section 6.4, ”throughput” in section 6.5 and ”responsiveness” in section 6.6. Each group includes multiple benchmarks in which we changed on variable at a time. The results are presented directly after each benchmark followed by a discussion to interpret the results.

In chapter 7 we draw a conclusion over out work and give the answer to our question from above. Also ideas for future research and development in this field are presented.

Finally we in in chapter 8 with a short summary.

# 2. Background & Related Work

## 2.1 Industrial Data

## 2.2 Graphs

## 2.3 Graph Databases

### 2.3.1 Triple Stores

#### 2.3.1.1 Apache Jena

### 2.3.2 Document Stores

#### 2.3.2.1 OrientDB

### 2.3.3 Graph Stores

#### 2.3.3.1 Neo4j

#### 2.3.3.2 Sparksee

## 2.4 Graph Database Benchmarks

### 2.4.1 LDBC: Graphalytics

### 2.4.2 XGDBench

### 2.4.3 YCSB

### 2.4.4 ConclusionTODO: show features to compare

## 2.5 Related Work

### 2.5.1 Graph Database: Anna

### 2.5.2 TODO: Add more

# 3. Analysis

## 3.1 Data

### 3.1.1 Data StructureNOTE: Here or in Design?

### 3.1.2 Data Amount

## 3.2 Workloads

### 3.2.1 Inserting Data into the Database

NOTE: What is the pattern of insertion

### 3.2.2 Retrieving Data from the Database

NOTE: What is the pattern of retrieving

## 3.3 Benchmark - YCSB

NOTE: Activity diagram in this section. What WAS the workflow.

# 4. Design

## 4.1 Data Structure

## 4.2 Workloads

### 4.2.1 Inserting

### 4.2.2 Production Simulation

### 4.2.3 Reading under load

## 4.3 Extension of the Benchmark

### 4.3.1 Generating a Dataset

#### 4.3.1.1 Storing the Dataset

#### 4.3.1.2 Restoring the Dataset

### 4.3.2 Graph Workload

NOTE: Graph Workload functionality NOTE: Activity diagram of the workflow with graph data.

### 4.3.3 Bindings

#### 4.3.3.1 Apache Jena

#### 4.3.3.2 Neo4j

#### 4.3.3.3 OrientDB

#### 4.3.3.4 Sparksee

# 5. Implementation of your Project

## 5.1 Graph Data Generator

### 5.1.1 Parameters

### 5.1.2 Graph Data Creator

### 5.1.3 Graph Data Recreator

## 5.2 Graph Database Bindings

NOTE: Highlight the mapping of data and other specialities

### 5.2.1 Apache Jena

### 5.2.2 Neo4j

### 5.2.3 OrientDB

### 5.2.4 Sparksee

## 5.3 Graph Workload

### 5.3.1 Parameters

# 6. Evaluation

## 6.1 Objective

## 6.2 Setup

### 6.2.1 Hardware

### 6.2.2 Software

## 6.3 ExecutionNOTE: Scripts to run all benchmarks successively

## 6.4 Maximum Load

### 6.4.1 Probing Node CountNOTE: Comparing indexed to not indexed

#### 6.4.1.1 Results

#### 6.4.1.2 Discussion

### 6.4.2 Probing Node SizeNOTE: See change over increasing node size

#### 6.4.2.1 Results

#### 6.4.2.2 Discussion

## 6.5 Throughput

### 6.5.1 Difference without Edges

#### 6.5.1.1 Results

#### 6.5.1.2 Discussion

### 6.5.2 Product ComplexityNOTE: More child nodes.

#### 6.5.2.1 Results

#### 6.5.2.2 Discussion

### 6.5.3 Production SuitabilityNOTE: Testing production like workload

#### 6.5.3.1 Results

#### 6.5.3.2 Discussion

## 6.6 Responsiveness

### 6.6.1 Reading under load

#### 6.6.1.1 Results

#### 6.6.1.2 Discussion

### 6.6.2 Scanning under load

#### 6.6.2.1 Results

#### 6.6.2.2 Discussion

# 7. Conclusion and Future Work

## 7.1 Conclusion

### 7.1.1 Suitability

### 7.1.2 General Performance of Databases

## 7.2 Future Work

### 7.2.1 More Bindings

### 7.2.2 Concurrency

### 7.2.3 Other input methodsNOTE: I only used native Java APIs, to directly test the database.

### 7.2.4 WorkloadsTODO: what kind?

# 8. Summary