HOBBIT

Collaborative Project

Holistic Benchmarking of Big Linked Data

Project Number: 688227 **Start Date of Project:** 2015/12/01 **Duration:** 36 months

Deliverable 7.3.1
First Challenge Results Overview

Dissemination Level	Public
Due Date of Deliverable	Month 26, 31/01/2018
Actual Submission Date	Month 26, 31/01/2018
Work Package	WP7 - Organization of Evaluation Campaigns
Task	T7.1 & T7.3
Type	Report
Approval Status	Final
Version	1.0
Number of Pages	51
Filename	D7.3.1_First_Challenge_Results_Overview.pdf

Abstract: This deliverable comprises the overview report of the first round of the HOBBIT challenges. The best technologies developed during these challenges are presented and the main outcomes and achievements are summarized.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 688227.

History

Version	Date	Reason	Revised by
0.1	10/01/2018	First draft created	Vassiliki Rentoumi & Grigorios Tzortzis (NCSR-D)
0.2	14/01/2018	Peer Reviewed	Gayane Sedrakyan (IMINDS)
0.3	17/01/2018	Content Revised	Vassiliki Rentoumi & Grigorios Tzortzis (NCSR-D)
1.0	18/01/2018	Final version created	Vassiliki Rentoumi & Grigorios Tzortzis (NCSR-D)

Author List

Organization	Name	Contact Information
NCSR-D	Grigorios Tzortzis	gtzortzi@iit.demokritos.gr
NCSR-D	Vassiliki Rentoumi	vrentoumi@iit.demokritos.gr
InfAI	René Speck	speck@informatik.uni-leipzig.de
AGT	Martin Strohbach	MStrohbach@agtinternational.com
AGT	Pavel Smirnov	PSmirnov@agtinternational.com

Executive Summary

This deliverable provides an overview report of the outcomes derived from the first round of the HOBBIT challenges. HOBBIT organized 5 challenges over the second project year. We present the best technologies developed during the 5 HOBBIT challenges and summarize the main results and achievements.

In the Introduction (Section 1) we provide an overview of the main ideas and goals of this deliverable. In Section 2 we describe the HOBBIT benchmarks that have been used within the HOBBIT challenges. In Sections 3, 4, 5, 6 and 7, we present a complete overview of the 5 challenges and their results. In Section 8 we summarize the achievements performed through the first series of the challenges. We also report the feedback provided from the participants and how we intend to integrate this in the second series which we have already started to organize.

Abbreviations and Acronyms

WP	Work Package
BLD	Big Linked Data
KPIs	Key Performance Indicators
IM	Instance Matching
OAEI	Ontology Alignment Evaluation Initiative
OKE	Open Knowledge Extraction
MOCHA	Mighty Storage Challenge
QALD	Question Answering over Linked Data
DEBS GC	Debs Grand Challenge
SQA	Scalable Question Answering
SML	Structured Machine Learning
SDK	Software Development Kit
StreaML	Stream Machine Learning
DSB	Data Storage Benchmark
SPBv	Versioning Benchmark
QA	Question Answering

Contents

List of Tables

List of Figures

1 Introduction

This deliverable comprises the results overview report of the first series of HOBBIT challenges. We present the best technologies developed during the first series of the challenges and summarize the main outcomes and achievements. Through the 5 HOBBIT challenges organized so far we aimed to measure the performance of technologies for the different steps of the Big Linked Data (BLD) lifecycle. In contrast to existing benchmarks, we managed to provide modular and easily extensible benchmarks for all industry-relevant BLD processing steps that allow to assess whole suites of software that cover more than one step. We also plan to make available the necessary infrastructure to run the evaluation campaigns.

A description of the HOBBIT benchmarks which participated in the first series of the HOBBIT challenges is presented in Section 2. In Sections 3, 4, 5, 6, 7, we present a complete overview of the results produced from each of the 5 HOBBIT challenges. In these sections we also show that participating systems demonstrated comparable results to the relevant state-of-the-art systems that participated as baselines in the 5 HOBBIT challenges. The majority of the participating systems together with their results have been described in papers submitted to the corresponding challenges. The papers were peer-reviewed by experts and the most promising systems were invited to participate. The systems were tested against a concrete set of evaluation criteria (a.k.a Key Performance Indicators (KPIs)) for each challenge that were chosen based on the defined experimental set-up for each challenge task. The evaluation criteria defined for each challenge are described in detail on the corresponding sections of each challenge and are further reported in D 9.2.2 – Annual Public Report of the Second Year.

More specifically in the Hobbit challenges the participating systems were compared with the state-of-the-art systems such as Virtuoso Open Source 7.2.4[1] in the MOCHA challenge, the FOX [24] system in the OKE challenge, and ganswer2 [31] in QALD challenge. Most of the times the participating systems produced either comparable or better results to those of the state-of-the-art systems.

Additional information on the first series of HOBBIT challenges can be found on the project's website[2], as well as in related deliverables: D7.1.1 – First Workshop Proceedings, D7.2.1 – First Workshop Organization Report and D7.4.1 – First Challenge Evaluation. D7.1.1 reports on the proceedings of the challenges, D7.2.1 reports on the organizational aspects of the challenges and D7.4.1 reports on the quantitative and qualitative evaluation of the challenges.

In Section 8 we summarize the results obtained through the first series of the challenges. We also report the feedback provided from the participants and how we intend to address this in the second series of the challenges which we have already started to organize.

2 Benchmarks

Eight benchmarks have been developed within the HOBBIT project. A summary of these benchmarks is given in Table 1. All these benchmarks have been employed in the five HOBBIT challenges which have already been successfully performed. Table 2 shows a mapping between the HOBBIT benchmarks and of the challenges in which the latter participated.

The following subsections aim to give an understandable overview of these benchmarks with only the necessary technical details.

[1] https://github.com/openlink/virtuoso-opensource/tree/stable/7
[2] https://project-hobbit.eu/

Table 1: Overview of HOBBIT Benchmarks.

Benchmark	Short Description
Data Acquisition	Evaluate storage solutions that deal with the ingestion of streams of RDF data
Knowledge Extraction	Test the performance (runtime and accuracy) of entity recognition and linking frameworks over streams of unstructured data (text)
Link Discovery	Go beyond mere instance matching and check how well tools performs on other types of links (e.g., geospatial links) when faced with large amounts of data
Structured Machine Learning	Study the performance of machine Learning techniques (i.e., performance and runtime) on streams of structured data (e.g., RDF)
Data Storage	Stress test storage solutions for RDF when faced with realistic scenarios such as being the backend of a social network
Versioning	Check how well storage solutions deal with storing evolving data available in several versions and performing queries on and across these different versions
Question Answering	Evaluate the performance of data access solutions that can answer questions in natural language as well as keyword queries on large amounts of data
Faceted Browsing	Test storage solutions w.r.t. their performance as backends of data browsers

Table 2: Mapping of HOBBIT Challenges/Benchmarks.

Event	Benchmark
MOCHA @ESWC 2017 May 28th to June 1st, 2017	Data Acquisition (INFAI) Data Storage (OpenLink) Versioning (FORTH) Faceted Browsing (IAIS)
OKE @ESWC 2017 May 28th to June 1st, 2017	Knowledge Extraction (INFAI)
QALD @ESWC 2017 May 28th to June 1st, 2017	Question Answering (IAIS)
Grand Challenge @DEBS 2017 June 19th to June 23rd, 2017	Structured Machine Learning (AGT)
Link Discovery Track OM@ISWC 2017 October 21st to October 25th, 2017	Link Discovery (FORTH)

2.1 Data Acquisition

The constant growth of data in velocity and volume has increased the need to integrate and process data using efficient and scalable storage approaches. The task of a storage system is two-fold: (1) retrieve and store the data and (2) process multiple users questions (queries) in parallel. In most real-time applications, such as financial transactions or predictive maintenance, both tasks must be completed in parallel with as small a latency as possible. The aim of this benchmark is to measure the performance of such systems in terms of efficiency and completeness when faced with streams of input data. To achieve this goal, we study the behavior of existing systems when faced with data of increasing volume and velocity. In order to emulate a realistic scenario, the data is generated from one or multiple resources and is inserted into a storage system simultaneously. Then, queries are used to test the system's ingestion performance and storage abilities.

The current status of the benchmark system for data ingestion is as follows:

1. Data is obtained from one resource in the form of statements and is ordered and stored in files using a time stamp. The time stamp indicates the point of time that a statement was generated.
2. Statements with the same time stamp are inserted into a storage system simultaneously. The data ingestion between statements of different time stamps is delayed by a dilatation factor. The dilatation factor decreases as more sets of statements are inserted into the system. The final set of statements with the same time stamp has a dilatation value of 0.
3. After a set of statements of different time stamps are inserted into the storage system, a query is performed against the system. The goal of this query is to check if the last statement was successfully inserted.
4. The emulation stops when there are no more statements to be inserted.

The benchmark was used in the MOCHA challenge [7] and was able to unveil the limitations of existing systems pertaining to data ingestion. The following tasks are foreseen for the upcoming version of the benchmark:

1. We will introduce dependencies between data resources by creating a network among them.
2. We will test the ingestion performance of a storage system by deploying datasets that vary in volume (size of statements and time stamps).
3. We aim to use dilatation factors based on the real time differences between the various statement, in order to benchmark the system within a particular time interval.
4. We shall use streaming data from multiple resources.
5. We will vary the queries to cover different proportions of inserted statements.

More details about the Data Acquisition benchmark can be found in https://ckan.project-hobbit.eu/dataset/benchmark-for-sensor-data-odin.

2.2 Knowledge Extraction

A considerable portion of the information in data on the Web is still only available in unstructured form, i.e., without predefined formal structure. The goal of this benchmark is to evaluate how well knowledge acquisition frameworks for unstructured data perform. In particular, the benchmark will test the performance of systems that implement approaches for analyzing unstructured data streams (Twitter, RSS feeds, etc.).

A generic data generator that produces unstructured natural-language data streams was needed. Therefore, the work on this benchmark started with the analysis of natural language data streams (e.g., messages from social networks, content of crawled web pages). The analysis was based on the syntactical and semantic characteristics of this data. The different characteristics of the data streams were stored and used as input for the generic data generator. The generator produces natural language data streams that have a structure similar to that of the data on which it was trained, e.g, tweets.

The task generator for unstructured streams generates tasks aiming at the extraction of structured data from the given stream of unstructured data. These tasks can range from the recognition of known entities inside the text to the extraction of new, unknown entities and their properties. Every type of task generated by the task generator for unstructured data is associated with a set of performance indicators (e.g. recall, precision, F-measure, throughput) that has to be computed to evaluate the generated results. Thus, every task generator is connected to an evaluation module that can calculate this set of key performance indicators based on the generated result and a given gold standard.

We performed the following in the first year:

- GERBIL reuse for this task.
- Mimicking Twitter streams.
- Verbalization of knowledge base facts.
- Prototype integration of the benchmark into the HOBBIT platform.

The result was the foundation of the Open Knowledge Extraction Challenge at ESWC 2017 [25]. In the second year, we performed the following extensions to our previous paradigm:

- Integration of extraction benchmarks into the HOBBIT platform.
- Co-organization of the OKE Challenge at ESWC 2017.
- Dockerization of the benchmarks.
- Run the benchmarks.

We aim for the following goals in the third year:

- Extend the task generator.
  - Benchmark for property extraction between entities.
  - Benchmark for knowledge extraction.
- Extend the data generator for this new benchmarks.
- Integration of key performance indicators for this new benchmarks.

- Run the benchmarks and co-organize a corresponding challenge.

More details about the Knowledge Extraction benchmark can be found in https://ckan.project-hobbit.eu/dataset/data-extraction-benchmark-for-unstructured-data.

2.3 Link Discovery

A number of real and synthetic benchmarks that address different data challenges have been proposed for evaluating the performance of link discovery systems. So far, only a limited number of link discovery benchmarks target the problem of linking geo-spatial entities. However, some of the largest knowledge bases on the Linked Open Data Web are geo-spatial knowledge bases (e.g., LinkedGeoData). Due to the large amount of available geo-spatial datasets employed in Linked Data and in several domains, it is critical that benchmarks for geo-spatial link discovery are developed.

During the second year of the HOBBIT project we completed the implementation of two benchmark generators that deal with link discovery for spatial data:

- the *Linking Benchmark* generator based on SPIMBENCH [18], to test the performance of Instance Matching tools that implement string-based approaches for identifying matching spatial entities and the
- *Spatial Benchmark* generator that can be used to test the performance of systems that deal with topological relations proposed in the state of the art DE-9IM (Dimensionally Extended nine-Intersection Model) model [26].

Both benchmarks are generic in the sense that they are *schema agnostic*: they can operate with any datasets that contain trajectories, a trajectory being a set of points or a set of longitude, latitude pairs. For both benchmarks we used the TomTom datasets provided in the context of project HOBBIT.

The first benchmark is simple and can be used not only by instance matching tools, but also by SPARQL engines that deal with query answering over geo-spatial data. The second is more complex and implements all topological relations of DE-9IM between trajectories in the two dimensional space. Both benchmarks follow a *choke-point*-based design with the aim to address the technical difficulties that the different systems must solve.

The choke points for the first benchmark are a subset of the ones that were used for the development of SPIMBENCH. The ontologies used to represent trajectories are fairly simple, and do not consider complex RDF or OWL schema constructs already supported by SPIMBENCH. Nevertheless, since the Linking Benchmark is based on SPIMBENCH, in the case in which the ontology is complex, introducing semantics-aware modifications is straightforward. The test cases implemented in the benchmark focus on string-based transformations with different levels, types of spatial object representations and types of date representations. Furthermore, the benchmark supports addition and deletion of ontology (schema) properties, known also as schema transformations. The datasets that implement those test cases can be used by Instance Matching tools to identify matching entities. In a nutshell, the benchmark can be used to check whether two traces with their points annotated with place names designate the same trajectory. The Linking Benchmark gets as input, a dataset that consists of various traces, a trace being a sequence of points. The points are expressed using standard longitude/latitude coordinates and annotated with place names obtained from external Linked Data sources such as Google Maps and Foursquare.

For the design of the Spatial Benchmark generator, we focused on (a) on the correct implementation of all the topological relations of the DE-9IM topological model and (b) on producing large datasets, large enough to stress the systems under test. To the best of our knowledge, there exist few systems

that implement all the topological relations of DE-9IM, hence the benchmark already addresses the first choke point set. Moreover, we produced very large synthetic datasets using TomTom's original data and mimicking algorithm, and hence we are able to challenge the systems regarding dataset scale. The benchmark gets as input a set of traces, which consists of various traces, each trace being a sequence of points. The points are expressed using standard longitude/latitude coordinates. In the Spatial Benchmark, we considered that the traces are represented in the Well-known text (WKT) format. Appropriate transformations are applied to the input set of traces in order to obtain the target dataset that can be used to test the ability of systems to identify DE-9IM topological relations. The gold standard is produced after the generation of the source and target datasets.

Both benchmarks were successfully used in two tasks of the Ontology Alignment Evaluation Initiative (OAEI) that was held at ISWC 2017. More specifically we organised the *HOBBIT Track* with two tasks, the *Linking* and *Spatial* each running the corresponding HOBBIT benchmarks. The aim of the Track was to test the performance of Link Discovery tools that implement string-based as well as topological approaches for identifying matching spatial entities. The different frameworks were evaluated for both accuracy (precision, recall and F-measure) and time performance. Four systems participated in the Track and the results of the systems appear in https://project-hobbit.eu/challenges/om2017/ (tab Results).

We are currently working on extending the Spatial Benchmark to deal with *polygons* in addition to *linestrings*. The benchmark can therefore be used to test systems that answer queries regarding the relation of lines with polygons (i.e., does a road cross a city) and polygons with polygons (i.e., is a hotel built in a Natura designated area).

More details about the Link Discovery benchmark can be found in https://ckan.project-hobbit.eu/dataset/linkingbenchmark.

2.4 Structured Machine Learning

The value of machine learning in a modern world cannot be overstated. Many applications that we are using on a daily basis incorporate some machine learning components. The ability to benchmark such components in a reliable and reproducible way will improve their quality which will not go unnoticed. The availability of a structured data has increased over the past years. The structure can be seen as a background knowledge — an additional input to a learner providing some insight into the data. The Structured Machine Learning (SML) Benchmark developed in the HOBBIT project incorporates a certain type of machine learning that operates on a structured data.

During the second year of the project we were focused on the implementation SML Benchmark and evaluation of it the at the ACM DEBS GrandChallenge 2017 [3] and receiving a feedback. The following subtasks have been completed:

- SML benchmark was implemented and integrated into the HOBBIT platform. The benchmark is focused on comparison the performance of distributed stream processing analytical systems by measuring the correctness of found anomalies, latency of the system (how long does it take to send a response), and throughput (how much data can be processed in a second).
- Evaluation of the developed benchmark was done via organization of the DEBS Grand Challenge 2017 on the HOBBIT platform. Benchmarked systems had to process over 5GB of data that was sent to the systems under test at a rate of 250 data points second. In total 14 teams registered to participate and 7 teams correctly identified the anomalies.

[3] http://lsd1.ls.fi.upm.es/debs2017/call-for-grand-challenge-solutions/

- Initial preparations for SML Benchmark v2.0 have been done. The dataset and queries are related to marine traffic domain, accuracy and speed are chosen as KPIs. The evaluation of the SML Benchmark v2.0 will be organized at the upcoming DEBS Grand Challenge 2018 [4] .
- A set of generic abstractions have been distinguished from source codes of SML benchmark and published[5] as a standalone Software Development Kit (SDK). The SDK is focused on runtime orchestration of docker containers and allows to execute/debug the benchmarks/benchmarking systems using them either "as is" (and hit the breakpoints in the code) or being packed into docker containers (the same manner as components will be operated by the online platform). As a result it helps platform users to design and develop new benchmarks and benchmarking systems with less efforts. The SDK will be used for implementation of SML Benchmark v2.0 and will be introduced to the upcoming DEBS Grand Challenge participants.

For the third year of the project we are planning the following subtasks:

- To implement the SML Benchmark v2.0 and integrate it to the HOBBIT platform.
- To implement prototype benchmarking system to use it as baseline for upcoming challenge evaluation.
- To investigate the performance characteristics and bottlenecks of the components in the SML benchmark v2.0 (e.g. between the task generator and benchmarking system).
- To organize the upcoming DEBS Grand Challenge 2018 with use of the SML Benchmark v2.0 and receive a feedback, which is planned to be used for further improvement of the benchmark.

More details about the Structured Machine Learning benchmark can be found in https://ckan.project-hobbit.eu/dataset/sml.

2.5 Data Storage

In recent years, the huge expansion of the Linked Data Web in data volume has increased the need for triple stores to process more and more data in a shorter time. The systems should be able to handle a growing amount of triples and show its potential how they can be enlarged in order to accommodate that growth. Some of the applications request fast and reliable responses from data stores, usually in an interactive time (less than a second), regardless of the scale of dataset.

As a starting point for our Data Storage benchmark (DSB), we used LDBC Social Network Benchmark developed in LDBC project[6]. Workloads are designed to mimic the different usage scenarios found in operating a real social network site. Each workload defines a set of queries and query mixes, designed to stress the systems under test in different choke-point areas, while being credible and realistic. In previous two years, we focused on the Interactive workload, which reproduces the interaction between the users of the social network by including lookups and transactions that update small portions of the data base. These queries are designed to be interactive and target systems capable of responding such queries with low latency for multiple concurrent users.

[4] http://www.cs.otago.ac.nz/debs2018/
[5] http://github.com/hobbit-project/java-sdk
[6] http://www.ldbcouncil.org/

The benchmark was used as a task in the MOCHA challenge [7], held at ESWC 2017[7], as well as in the Open Challenge[8] which is currently in progress.

As promised, in the second year of the project we finished a considerable number of tasks, from which we emphasize the following:

- Modifications of the queries, related to the new version of data generator: A software that contains the mimicking algorithm responsible for generation of the data needed by the benchmark has been changed in the first year of the project, and now, the queries are adopted to be suitable for the newly introduced features, predicates and distributions.
- Dockerization of the benchmark and porting on HOBBIT platform: DSB v1.1 has been fully integrated into the HOBBIT platform, and all interested parties can evaluate their system against it by running the benchmark through the platform website.
- Common API: Our benchmark shares the agreed and introduced API with couple of other benchmarks.
- Experiments with the full benchmark on different scale factors: DSB has been part of two challenges where different systems has been tested against it, bringing the results of their performance.
- Local deployment of the platform used for internal testing.

Future work includes:

- Implementation of the second version of the benchmark (DSB v2.0): Unlike the first version that was single threaded, this one will be multi-threaded, allowing us to test the concurrency of the systems using a real workload that will be driven by updates.
- Mutual comparisons between different scales, and with SQL implementation of the benchmark.
- Further evaluations, etc.

More details about the Data Storage benchmark can be found in https://ckan.project-hobbit.eu/dataset/data-storage-benchmark.

2.6 Versioning

The open nature of the Web implies that a number of changes typically happen without any warning, centralized monitoring, or reliable notification mechanisms. This raises the need to keep track of the different *versions* of the datasets and introduces new challenges related to assuring the quality and evolution of Web data over time.

The objective of this task is to propose a synthetic and scalable benchmark based on LDBC's[9] SPB 2.0 benchmark[10] to test the ability of systems to store and query different versions of an evolving dataset. In the second year of the project we performed the following tasks:

[7] https://project-hobbit.eu/challenges/mighty-storage-challenge/
[8] https://project-hobbit.eu/open-challenges/mocha-open-challenge/
[9] http://ldbcouncil.org
[10] http://ldbcouncil.org/benchmarks/spb

- We finalized the implementation of the first version of our benchmark **SPBv** v1.0. In particular, we finalized the implementation of the benchmark's *data generator* that started during the first year of the project and continued with the implementation of the *query workload* that consisted of queries of eight different types that included some of the LDBC's SPB 2.0 benchmark original queries.
- We ported the versioning benchmark into the HOBBIT platform.
- We conducted a set of experiments for testing **SPBv** v1.0 on top of the HOBBIT platform, using two different triple stores. The first one was Virtuoso Open source 7.2.4[11], in which we implemented the *full materialization* archiving strategy and the second one was the pure *versioning system* R43ples[12].
- We designed and started the implementation of the second version of our benchmark (**SPBv** v2.0). More specifically, we designed and started to implement the data generator that supports in addition to *additions of triples* in the new version(s), *deletions* of already existing instances (i.e., creative works that are metadata descriptions of journalistic assets). Furthermore, we enhanced the generated instances with five different versions of the DBpedia entities that are used for their annotation. By having DBpedia data in our dataset, the query workload of (**SPBv** v2.0) can be based on real query logs from DBpedia.
- **SPBv** v1.0 was used in the MOCHA open challenge[13] that has already been launched.

During the third year of the project we will proceed with the:

- Finalization of the implementation of the data generator for the **SPBv** v2.0 that will support addition and deletion of triples to create the different versions.
- Finalization of the query workload for the second version of **SPBv** that will be based on real DBpedia query logs. The DBpedia query logs will be used to define templates for the **SPBv** queries that will be instantiated with values from the produced datasets. The choice of the parameters will be based on criteria such as selectivity in order to produce a well balanced workload.
- Implementation of necessary system adapters that will be used to integrate versioning systems in the HOBBIT platform that will be subsequently benchmarked with **SPBv** v2.0.

More details about the Versioning benchmark can be found in https://ckan.project-hobbit.eu/dataset/versioning-benchmark.

2.7 Question Answering

The past years have seen a growing amount of research on question answering over large-scale RDF data. At the same time, the growing amount of data has led to a heterogeneous data landscape. The purpose of this work package is to provide up-to-date benchmarks for assessing performance and accuracy of question answering approaches that mediate between users, expressing their information need in natural language, and large-scale background knowledge data (i.e. DBpedia). The evaluation platform for this benchmarks was built upon the open source GERBIL QA benchmarking platform[14]. Concerning question answering tasks, the general task is:

Given one or several RDF datasets as well as additional knowledge sources and natural language questions or keywords, return the correct answers or a SPARQL query that retrieves these answers.

[11] https://github.com/openlink/virtuoso-opensource/tree/stable/7
[12] https://github.com/plt-tud/r43ples
[13] https://project-hobbit.eu/open-challenges/mocha-open-challenge/
[14] http://gerbil-qa.aksw.org/gerbil

Our specific benchmarks include:

- multilingual question answering over DBpedia, i.e. such that answers can be retrieved from an RDF data repository given an information need expressed in a variety of languages (including English, German, Dutch, French, Spanish, Italian, Romanian, Persian and Hindi);
- hybrid question answering, requiring the integration of both RDF and textual data sources;
- large-scale question answering, including a massive amount of automatically derived questions taking in to account not only the accuracy of a system's answers but also the time needed to retrieve those answers.

So far we have:

- Completed in-depth investigations on the main challenges of question answering over linked data.
- Created new datasets, one of which massive, to better enable the benchmarking of systems according to the QA challenges identified.
- Incorporated the first version of the QA Benchmark into the HOBBIT platform.
- Participated in the QALD-7 challenge in ESWC2017 and in the QALD-8 challenge in ISWC2017.

In the third year of this project we will:

- Create and support ongoing open challenges, with cut-of dates coinciding with major conferences and meetings (please see below).
- Refresh the benchmark datasets and improve user and technical documentation.
- Improve the multilingual benchmark, by leveraging localised versions of DBpedia.
- Improve the large scale benchmark, by including complex and more varied questions.

More details about the Question Answering benchmark can be found in https://ckan.project-hobbit.eu/dataset/question-answering-benchmark.

2.8 Faceted browsing

Faceted browsing stands for a session-based and state-dependent interactive method for query formulation over a multi-dimensional information space. It provides a user with an effective way for exploration through a search space. After having defined the initial search space, i.e., the set of resources of interest to the user, a browsing scenario consists of applying (or removing) filter restrictions of object-valued properties or of changing the range of a number-valued property.

As a well-established example for an implementation of faceted browsing consider an online shopping portal where the search space could be a certain type of clothes and, amongst others, the facets could consist of size, color and price. Using the mentioned filtering operations aimed to select items with desired properties, the user browses from state to state, where a state consists of the currently chosen facets, their corresponding facet values and the current set of instances satisfying all chosen constraints.

The goal of the task on faceted browsing is to check existing solutions for their capabilities of enabling faceted browsing through large-scale structured datasets, that is, it analyses their efficiency

in navigating through large datasets, where the navigation is driven by intelligent iterative restrictions. We develop browsing scenarios through a dataset, which reflect an authentic use-case and challenge participating systems on different points of difficulty. To distinguish several solutions, we measure the performance relative to dataset characteristics, such as overall size and graph characteristics.

The following is the current state of this project task.

(i) We collected the choke point of faceted browsing, that is, the difficulties that arise for a system in enabling efficient browsing through structured datasets.

(ii) We explored our possibilities to develop benchmarking scenarios which resemble realistic browsing scenarios.

(iii) We investigated the characteristics of several datasets for their suitability to benchmark systems according to the choke points collected in (i).

The next steps are as follows:

- We will analyse existing faceted browsing engines for the types of SPARQL queries they generate and incorporate selected ones into the benchmark.
- We will add support for composing faceted browsing benchmarks from high-level user interaction events, such as panning a map or enabling/disabling certain constraints. This will increase the flexibility of the system by enabling benchmarking of different strategies that convert interactions to a (set of) SPARQL queries. For example, the same sequence of interactions can be benchmarked using a strategy that requests exact facet counts as well as one that allows for approximations.
- For all newly introduced queries / query types, we will classify them according to the identified choke-points and extend this list as necessary.
- We integrate the revised benchmark into the HOBBIT platform.

More details about the Faceted Browsing benchmark can be found in https://ckan.project-hobbit.eu/dataset/faceted-browsing-benchmark.

3 MOCHA Challenge Results Overview

3.1 Definition of Tasks

Through the Mighty Storage Challenge (MOCHA) we aimed at testing the performance of solutions for SPARQL processing in aspects that are relevant for modern applications. Therefore, we aimed to benchmark systems that dealt with the benchmarks presented in Sections 2.1, 2.5, 2.6 and 2.8. These include ingesting data, answering queries on large datasets and serving as backend for applications driven by Linked Data. The challenge tested the systems against data derived from real applications and with realistic loads, putting emphasis on dealing with changing data in the form of streams or updates. MOCHA in 2017 defined three tasks:

- Task 1: Ingestion of RDF data streams

- Task 2: RDF data storage
- Task 3: Browsing RDF data

In more detail, Task 1 focused on measuring how well systems can ingest streams of RDF data, while Task 2 on measuring how data stores perform with different types of queries. Finally, Task 3 was about checking existing solutions for how well they support applications that need to browse through large datasets.

MOCHA was successfully held at ESWC 2017[15]. Systems participating in MOCHA as well as their results were presented to the public in a dedicated workshop session. The papers describing the systems of the MOCHA challenge were peer-reviewed by experts and the most promising systems were invited to participate in the challenge. After the reviewing process two systems were accepted to the challenge, on top of the baseline system provided by the challenge organizers.

The challenge papers have been published by Springer on the proceedings volume *Dragoni M., Solanki M. and Blomqvist E. (eds), Semantic Web Challenges, Communications in Computer and Information Science, vol. 769, 2017*[16]. This volume contains the papers of all challenges that were organized at the ESWC 2017 conference. Specifically, in [7] the challenge organizers presented an overview of the challenge results and baseline systems, while in [15] and [23] the challenge participants described their systems.

3.2 Participating Systems

Three systems participated in MOCHA with all three teams addressing all tasks. One out of the three systems served as the baseline system.

Virtuoso Open-Source Edition 7.2 [7] [17], developed by OpenLink Software, served as the baseline system for all MOCHA2017 tasks (*MOCHA Baseline*). Virtuoso Open-Source Edition is a scalable cross-platform server that combines Relational, Graph, and Document Data Management with Web Application Server and Web Services Platform functionality.

QUAD [15][18], developed by Ontos. QUAD is a native RDF store based on a vector database schema for quadruples and it is realized by facilitating various index data structures. QUAD also includes approaches to optimize the SPARQL query execution plan by using heuristic transformations.

Virtuoso Commercial Edition 8.0 (beta) [23] [19], developed by OpenLink Software. Virtuoso Commercial Edition comprises a modern enterprise-grade solution for data access, integration, and relational database management, which provides a scalable RDF Quad Store.

3.3 Results & Achievements

3.3.1 Task 1

3.3.1.1 KPIs fo Task 1

The main KPIs for Task 1 were the following three:

[15] https://project-hobbit.eu/challenges/mighty-storage-challenge/
[16] https://doi.org/10.1007/978-3-319-69146-6
[17] https://virtuoso.openlinksw.com/
[18] http://ontos.com/
[19] https://virtuoso.openlinksw.com/

- Recall, Precision and F-measure: The INSERT queries created by each data generator were sent into a triple store by bulk load. After a stream of INSERT queries was performed against the triple store, a SELECT query was conducted by the corresponding data generator. In Information Retrieval, recall and recision were used as relevance measurements and were defined in terms of retrieved results and relevant results for a single query. Recall is the fraction of relevant documents that were successfully retrieved and precision is the fraction of the retrieved documents that are relevant to a query. F-measure is the harmonic mean of Recall and Precision. For our set of experiments, the relevant results for each SELECT query were created prior to the system benchmarking by inserting and querying an instance of the Jena TDB storage solution.

  Additionally, the following evaluation metrics were computed:

$$Macro{-}Average{-}Precision = \frac{\sum_{i=1}^{\lambda} Precision_i}{\lambda} \tag{1}$$

$$Macro{-}Average{-}Recall = \frac{\sum_{i=1}^{\lambda} Recall_i}{\lambda} \tag{2}$$

  where λ is the number of SELECT queries performed against the storage solution during the execution of the benchmark and Micro and Macro-Average Recall, Precision and F-measure of the whole benchmark. The aforementioned measurements $Precision_i$ and $Recall_i$ are the precision and recall of the i-th SELECT query. We also calculated $Macro{-}Average{-}F{-}measure$ as the harmonic mean of Equations 1 and 2.

$$Micro{-}Average{-}Precision = \frac{\sum_{i=1}^{\lambda} |\{relevant\ results_i\} \cap \{retrieved\ results_i\}|}{\sum_{i=1}^{\lambda} |\{retrieved\ results_i\}|} \tag{3}$$

$$Micro{-}Average{-}Recall = \frac{\sum_{i=1}^{\lambda} |\{relevant\ results_i\} \cap \{retrieved\ results_i\}|}{\sum_{i=1}^{\lambda} |\{relevant\ results_i\}|} \tag{4}$$

  where the $\{relevant\ results_i\}$ and $\{retrieved\ results_i\}$ are the relevant and the retrieved results of the i-th SELECT query resp. We also calculated $Micro{-}Average{-}F{-}measure$ as the harmonic mean of Equations 3 and 4.

  Misclassifications between the expected and received results does not necessarily mean that the triple stores are prone to misclassify results or to have a bad performance, but that there are mismatches for results sets between Jena TDB and the storage solution.

- Triples per second: at the end of each stream and once the corresponding SELECT query was performed against the system, we measured the triples per second as a fraction of the total number of triples that were inserted during that stream. This was divided by the total time needed for those triples to be inserted (begin point of SELECT query - begin point of the first INSERT query of the stream). We provided the maximum value of the triples per second of the whole benchmark. The maximum triples per second value was calculated as the triples per second value of the last stream with Recall value equal to 1.

- Average answer time: we reported the average answer delay between the time stamp that the SELECT query has been executed and the time stamp that the results are send to the evaluation storage. The first aforementioned time stamp was generated by the benchmark when the SELECT query was sent to the system and the second time stamp was generated by the platform when the results of the corresponding SELECT query were sent to the storage.

3.3.1.2 Experiment set-up

ODIN require a set of parameters to be executed, that are independent of the triple store. For MOCHA, all three systems were benchmarked using the same values. Each triple store was allowed to communicate with the HOBBIT [11] platform for at most 25 $mins$. The required parameters and their corresponding values for MOCHA are:

- **Duration of the benchmark**: It determines the time interval of the streamed data. Value for MOCHA2017 = $600,000\,ms$.
- **Name of mimicking algorithm output folder**: The relative path of the output dataset folder. Value for MOCHA2017 = $output\_data/$.
- **Number of insert queries per stream**: This value is responsible for determining the number of INSERT SPARQL queries after which a SELECT query is performed. Value for MOCHA2017 = 100.
- **Population of generated data**: This value determines the number of events generated by a mimicking algorithm for one Data Generator. Value for MOCHA2017 = $10,000$.
- **Number of data generators - agents**: The number of indepedent Data Generators that send INSERT SPARQL queries to the triple store. Value for MOCHA2017 = 4.
- **Name of mimicking algorithm**: The name of the mimicking algorithm to be invoked to generate data. Value for MOCHA2017 = *TRANSPORT_DATA*.
- **Seed for mimicking algorithm**: The seed value for a mimicking algorithm. Value for MOCHA2017 = 100.
- **Number of task generators - agents**: The number of indepedent Task Generators that send SELECT SPARQL queries to the triple store. Value for MOCHA2017 = 1.

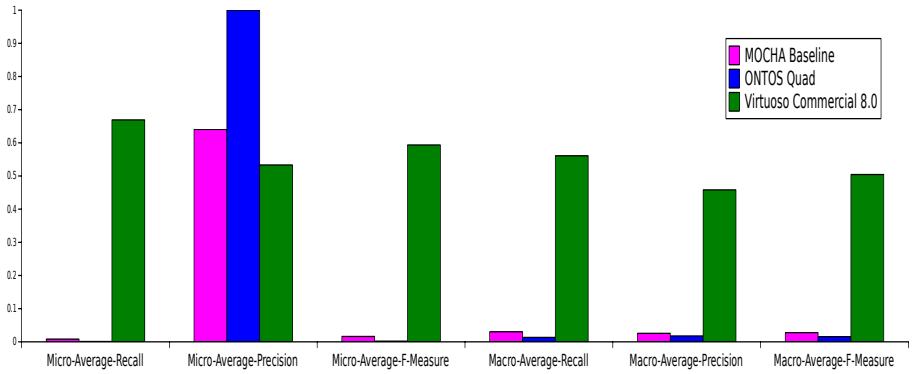


Figure 1: Micro-Average-Recall, Micro-Average-Precision, Micro-Average-F-measure, Macro-Average-Recall, Macro-Average-Precision, Macro-Average-F-measure of *MOCHA Baseline*, *ONTOS Quad* and *Virtuoso Commercial 8.0* for MOCHA2017.

3.3.1.3 Results for Task 1

Regarding Task 1 as it is shown in Figure 1, it is observed that *Virtuoso Commercial 8.0* has by far the best performance compared to the other two systems in terms of Macro and Micro-Average

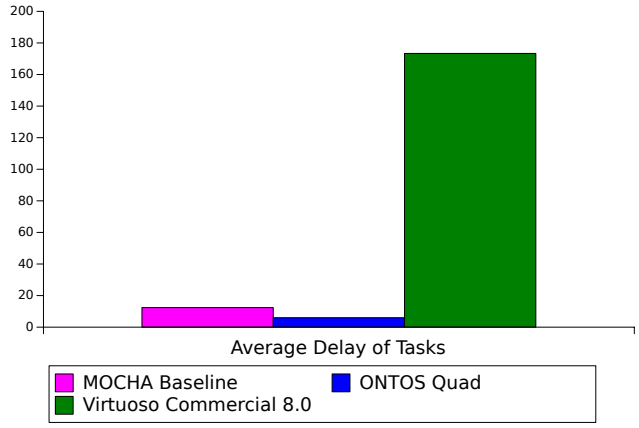


Figure 2: Average Delay of tasks of *MOCHA Baseline*, *ONTOS Quad* and *Virtuoso Commercial 8.0* for MOCHA2017.

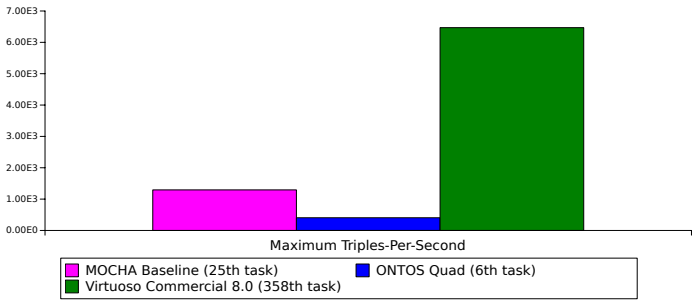


Figure 3: Maximum Triples-per-Second of *MOCHA Baseline*, *ONTOS Quad* and *Virtuoso Commercial 8.0* for MOCHA2017.

Precision, Recall and F-measure. *Virtuoso Commercial 8.0* was able to store and retrieve more triples throughout the whole benchmark. However, the maximum performance was achieved for Micro-Average Recall = 0.67, which indicates that the misclassifications between Jena TDB and *Virtuoso Commercial 8.0* were still high on average. Additionally, since the Micro-Average values were higher compared to the Macro-Average values, we can conclude by stating that *Virtuoso Commercial 8.0* was able to retrieve more relevant triples to a SELECT query, for tasks with higher quantity of expected results.

Figure 3 indicates that *Virtuoso Commercial 8.0* achieved the highest number of Triples-per-Second (TPS) at the latest task. It received the last recall value of 1 at task 358 (out of 395), whereas the other systems have issues with recall at much earlier stages of the benchmark. Especially for the *ONTOS Quad* system, we see that its recall drops significantly after the 6th SELECT query.

It is also worth mentioning that *ONTOS Quad* and *Virtuoso Commercial 8.0* were not able to perform all SELECT queries within 25 $mins$. *ONTOS Quad* was not able to send results to the evaluation storage throughout the whole benchmark, whereas *Virtuoso Commercial 8.0* was not able to execute SELECT queries after 358 tasks, which is one of the reasons why its recall drops to 0.

Finally, in Figure 2 task delay for each task for all systems is presented. We observed that all systems have a relatively low task average delay over the set of SELECT queries. Whereas *Virtuoso Commercial 8.0* has a monotonically ascending task delay function, that drops to 0 after the 358th task, since the system is no longer available because it exceeded the maximum allowed time to process queries.

3.3.2 Task 2

3.3.2.1 KPIs for Task 2

The main KPIs for Task 2 were:

- **Bulk Loading Time:** The total time in milliseconds needed for the initial bulk loading of the dataset.
- **Average Task Execution Time:** The average SPARQL query execution time.

- **Average Task Execution Time Per Query Type:** The average SPARQL query execution time per query type.
- **Number of Incorrect Answers:** The number of SPARQL SELECT queries whose result set is different from the result set obtained from the triple store used as a gold standard.
- **Throughput:** The average number of tasks executed per second.

3.3.2.2 Experiment set-up.

The Data Storage Benchmark has parameters which need to be set in order to execute the benchmark for this task. These parameters are independent of the triple store which is evaluated. The required parameters are:

- **Number of operations:** This parameter represents the total number of SPARQL queries that should be executed against the tested system. This number includes all query types: simple SELECT queries, complex SELECT queries and INSERT queries. The ratio between them, e.g. the number of queries per query type, has been specified in a query mix in such a way that each query type has the same impact on the overall score of the benchmark. This means that the simpler and faster queries are present much more frequently than the complex and slower ones. The value of this parameter for MOCHA was 15,000 operations.
- **Scale factor:** The DSB can be executed using different sizes of the dataset, i.e. with different scale factors. The scale factor for MOCHA was 1, i.e. the smallest DSB dataset.

3.3.2.3 Results for Task 2

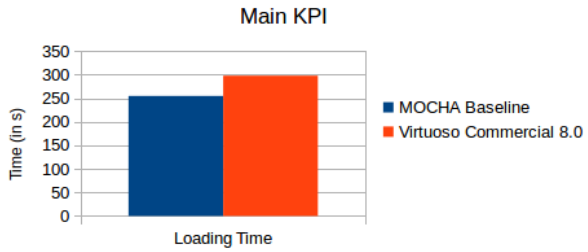


Figure 4: Loading Time

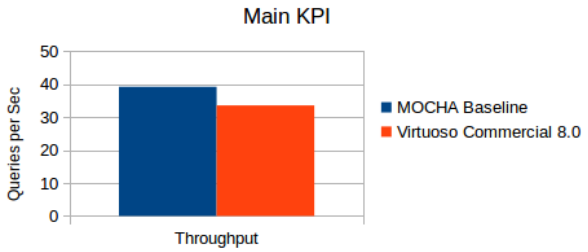


Figure 5: Throughput

Three systems were submitted for Task 2: Virtuoso 7.2 Open-Source Edition by OpenLink Software, Virtuoso 8.0 Commercial Edition (beta release) by OpenLink Software, and QUAD by Ontos. QUAD was not able to finish the experiment in the requested time (30 minutes), i.e. it exhibited a timeout.

Based on the results from the KPIs, shown in Figures 4, 5, 6 and 7, the winning system for the task was Virtuoso 7.2 Open-Source Edition by OpenLink Software.

Figure 6: Long Queries

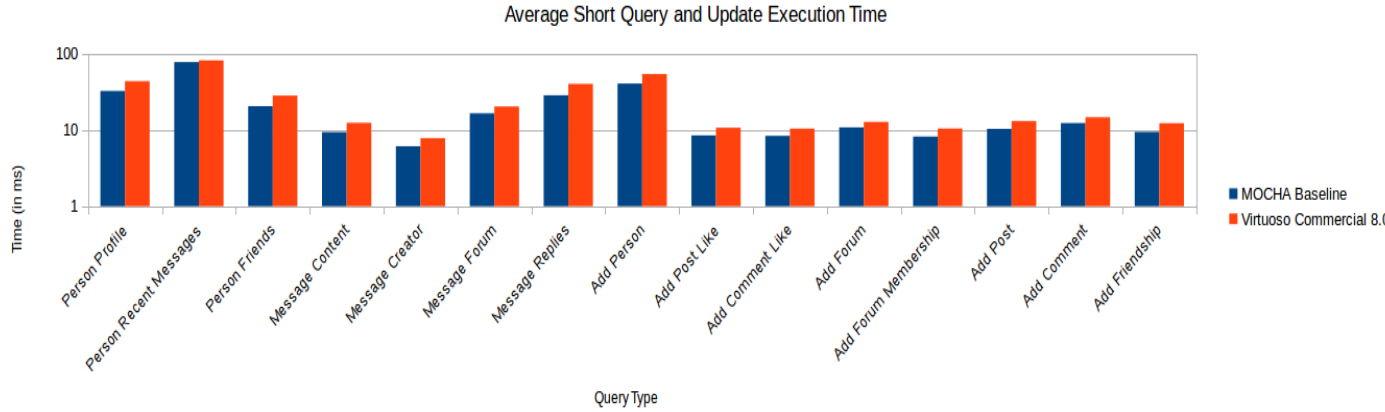


Figure 7: Short Queries and Updates

3.3.3 Task 3

3.3.3.1 KPIs for Task 3

For the evaluation, the received results from the participating system were compared with the expected ones. Results were returned in form of several key performance indicators:

The performance on instance retrievals was measured by a query-per-second score, by precision, recall and F1-score. Next to results for the full workload, the values were recorded for each of the 14 choke point individually. The list of choke points reads as follows:

1. Find all instances which (additional to satisfying all restrictions defined by the state within the browsing scenario) have a certain property value
2. Find all instances which (additionally) realize a certain property path with any value
3. Find all instances which (additionally) have a certain value at the end of a property path
4. Find all instances which (additionally) have a property value lying in a certain class

5. For a selected class that a property value should belong to, select a subclass
6. Find all instances that (additionally) have numerical data lying within a certain interval behind a directly related property
7. Similar to 6, but now the numerical data is indirectly related to the instances via a property path
8. Choke points 6 and 7 under the assumption that bounds have been chosen for more than one dimension of numerical data
9. Choke points 6,7,8 when intervals are unbounded and only an upper or lower bound is chosen
10. Go back to the instances of a previous step by unselecting previously chosen facets
11. Change the solution space to instances in a different class while keeping the current filter selections (Entity-type switch)
12. Choke points 3 and 4 with advanced property paths involved
13. Choke points 1 through 4 where the property path involves traversing edges in the inverse direction
14. Additional numerical data restrictions at the end of a property path where the property path involves traversing edges in the inverse direction

For facet counts, we measured the accuracy of participating systems in form of the deviation from the correct and expected count results. Additionally, we computed the query-per-second score for the corresponding COUNT queries.

3.3.3.2 Experiment set-up

The Faceted Browsing Benchmark required only one parameter which needed to be set in order to execute the benchmark for this task. This parameter consisted of a random seed, whose change alters the SPARQL queries of the browsing scenarios. The dataset was fixed and comprised about 1 million triples.

3.3.3.3 Results for Task 3

In Task 3 three systems were submitted: Virtuoso 7.2 Open-Source Edition by OpenLink Software which served as the MOCHA baseline, Virtuoso 8.0 Commercial Edition (beta release) by OpenLink Software, and QUAD by Ontos. Unfortunately, QUAD was not able to finish the experiment in the requested time (30 minutes), i.e. it exhibited a timeout. In Figure 8 and Figure 9, the results are shown on instance retrievals. It is detected that both systems experienced problems on choke point number 12, which corresponds to filtering for the realization of a certain property path (i.e., the task is to find all instances that, additionally to satisfying all restrictions defined by the state within the browsing scenario, realize a certain property path), and where the property path is of a rather complicated form. For instance, complicated paths include those containing circles, or property paths where multiple entries need to be avoided.

In Figure 9 the query-per-second score is presented. It is observed that the performance of both the open source and commercial versions of Virtuoso are very similar with a slight advantage for the open source version. Interestingly, the query-per-second score of both system is the lowest for choke

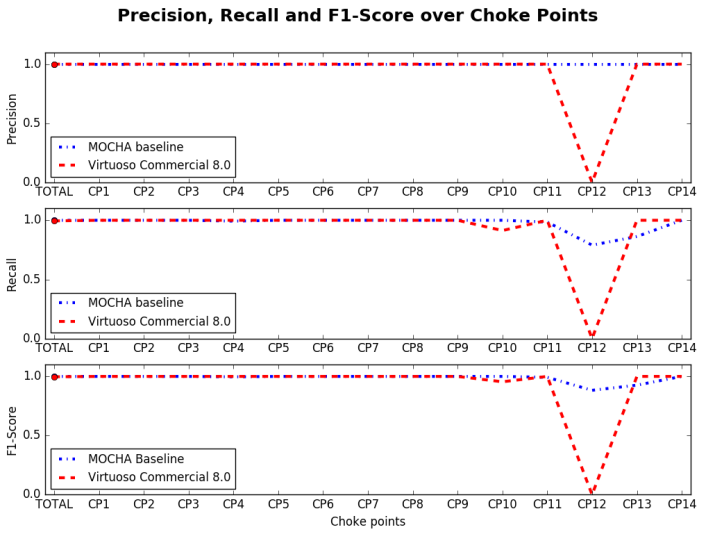


Figure 8: Instance Retrieval - Accuracy

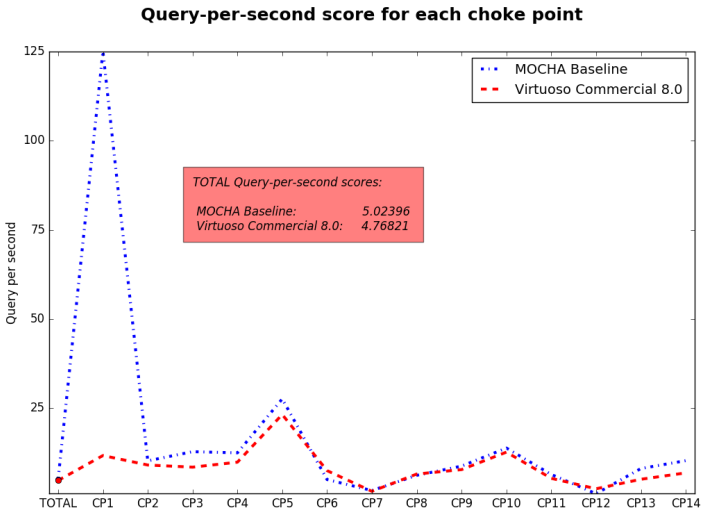


Figure 9: Instance Retrieval - Query-per-second score

points 6 -8, which all correspond to selections of numerical data at the end of a property or property path.

In Figure 10 and 11 the performance on count queries are shown. Again, the open source Virtuoso version had a slight advantage in the query-per-second score. On the other hand, the commercial version of Virtuoso made less errors in returning the correct counts. All in all both systems had very similar results with both having their slight advantage on one task or the other.

3.4 Conclusions

The successful organization of MOCHA in 2017 by HOBBIT will serve as the basis for the future editions of the challenge.

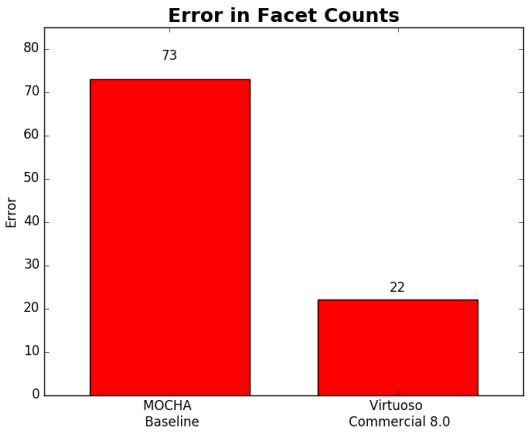


Figure 10: Facet Counts - Accuracy

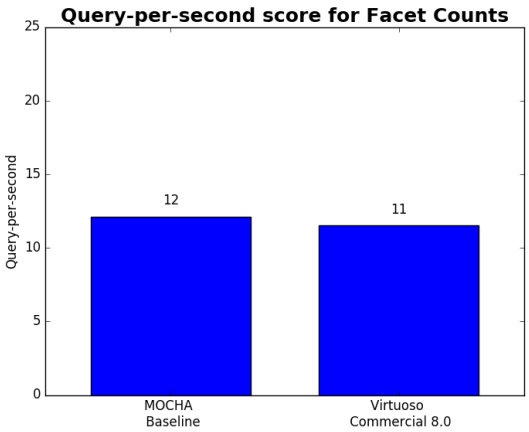


Figure 11: Facet Counts - Query-per-second score

We aim further simplify the participation process and offer leaderboards so that the challenge participants can easily preview their systems' performance. Overall, the MOCHA challenge's results suggest that while the scalability of triple stores is improving, a need for scalable distributed solutions remains. All the evaluation criteria and KPIs employed in all Tasks comprising the MOCHA challenge are further detailed and explained in [7].

Moreover, given the feedback from the participating teams, the HOBBIT MOCHA team intends to benchmark more triple storage solutions by scaling over the volume and velocity of the RDF data and use a diverse number of datasets to test the scalability of our approaches.

4 QALD Challenge Results Overview

4.1 Definition of Tasks

The Question Answering over Linked Data (QALD)[20] challenge aims at providing an up-to-date benchmark for assessing and comparing systems that mediate between a user, expressing his or her information need in natural language, and RDF data. It thus targets all researchers and practitioners working on querying Linked Data, natural language processing for question answering, multilingual information retrieval and related topics. The main goal is to gain insights into the strengths and shortcomings of different approaches and into possible solutions for coping with the heterogeneous and distributed nature of Semantic Web data. QALD in 2017 focused on the following four tasks:

- Task 1: Multilingual question answering over DBpedia
- Task 2: Hybrid question answering
- Task 3: Large-scale question answering over RDF
- Task 4: English question answering over Wikidata

In more detail, *Task 1* derives from the increasing need to facilitate multilingual access to semantic data, given the diversity of languages used on the web, and its target is to retrieve answers from an

[20] https://project-hobbit.eu/challenges/qald2017/

RDF data repository given an information need expressed in a variety of natural languages. *Task 2* focuses on the retrieval of answers for questions that require the integration of data both from RDF and textual sources, since a large amount of information is still available as unstructured text only and, therefore, approaches that combine information from structured and unstructured sources are needed. *Task 3* focuses on large-scale question sets with the aim to assess approaches that are able to scale up to a big data volume, handle a vast amount of questions and speed up the question answering process by parallelization, such that the highest possible number of questions can be answered as accurately as possible in the shortest possible time. *Task 4* contains questions originally formulated for DBpedia that require an answer using Wikidata, so that systems have to deal with a different data representation structure. Thus the task exploits the ability of systems to adapt to new data sources.

QALD was organized in conjunction with the ESWC 2017 conference, where systems participating in the challenge and their results were presented to the public in a dedicated workshop session. The papers describing the systems of the QALD challenge were peer-reviewed by experts and the most promising systems were invited to participate in the challenge. The challenge papers have been published by Springer on the proceedings volume *Dragoni M., Solanki M. and Blomqvist E. (eds), Semantic Web Challenges, Communications in Computer and Information Science, vol. 769, 2017*[21]. Specifically, in [27] the challenge organizers presented an overview of the challenge results and baseline systems, while in [6], [16] and [22] the challenge participants described their systems.

4.2 Participating Systems

Three teams/systems participated in QALD, with three teams addressing Task 1 (two for English and one for French) and two addressing Task 4. No systems were submitted for Tasks 2 and 3.

WDAqua [6] is a rule-based system using a combinatorial approach to generate SPARQL queries from natural language questions, leveraging the semantics encoded in the underlying knowledge base. It can answer questions on both DBpedia (supporting English) and Wikidata (supporting English, French, German and Italian). The system, which does not require training, participated in Tasks 1 and 4 of the challenge.

AMAL [16] has been developed for question answering in French. Firstly, the question type (e.g. *Boolean* or *Entity*) is classified by pattern matching. This induces the rerouting to the relevant question type solver where entities and properties are extracted: the former by syntactic parsing and subsequent linking to DBpedia entities; the latter by removing the found entity and searching for corresponding properties in DBpedia, possibly with the help of Wikipage disambiguation links. SPARQL predicate identification is supported by a manually curated lexicon of common DBpedia properties, each linked to one or more possible French expressions. The system can only answer simple questions (concerning a single entity or a single property of an entity) and participated in Task 1.

Sorokin and Gurevych [22] participated in Task 4 of the challenge. They provided a system producing the semantic representation of a natural language question, which is then deterministically converted into SPARQL. After minimal pre-processing, including POS tagging and entity linking, an end-to-end neural architecture employs a CNN neural scorer to choose among multiple semantic representations of the question. First, the semantic representations are generated by expansion on the knowledge base, guided by the entity found in the question and by all possible relations and constraints as present in the knowledge base for the entity. Then, each question and candidate representations are vectorialised, with the CNN producing comparison scores based on cosine similarity, leading to the final choice.

[21] https://doi.org/10.1007/978-3-319-69146-6

ganswer2 [31] participated (in Task 1) outside the actual challenge this year, as a baseline system without a paper submission to the challenge. ganswer2 uses a graph-based approach to generate a semantic query graph which reduces the transformation of natural language to SPARQL into a subgraph matching problem.

4.3 Evaluation Metrics

The systems participating in the challenge were evaluated with respect to precision, recall and F1-measure. For a question q, precision and recall are defined as follows:

$$\text{recall}(q) = \frac{\text{number of correct system answers for } q}{\text{number of gold standard answers for } q}$$

$$\text{precision}(q) = \frac{\text{number of correct system answers for } q}{\text{number of system answers for } q}$$

To assess the systems' performance over all the test questions included in each task, the macro and micro values for precision, recall and F1-measure were computed. For Task 3 the evaluation took into account not only the accuracy measures for the answered questions, but also the scalability measures in terms of number of processed queries and time needed for answer retrieval.

4.4 Results & Achievements

All tasks of the QALD challenge consisted of a training dataset that included questions along with their gold standard answers, and a testing dataset that did not contain gold standard answers for the questions. The systems participating in each of the challenge's tasks were evaluated both on the training and testing datasets using the HOBBIT platform.

Task 1 results are reported in Table 3. The detailed experimental results for Task 1 over training data can be found in the following links:

- ganswer (en): http://gerbil-qa.aksw.org/gerbil/experiment?id=201706300001
- AMAL (fr): http://gerbil-qa.aksw.org/gerbil/experiment?id=201706300002

For the testing data the results can be found in the following links:

- ganswer (en): http://master.project-hobbit.eu/#/experiments/details?id=1498647986590
- WDAqua (en): http://master.project-hobbit.eu/#/experiments/details?id=1498647742687
- AMAL (fr): http://gerbil-qa.aksw.org/gerbil/experiment?id=201706300011

Task 4 results are reported in Table 4. The two teams performed well on both the train and the test datasets and it can be observed that both systems have a higher macro F-measure than micro F-measure. Task 4 contains questions with long answer lists and if a system fails to answer such queries this has a huge impact on its micro recall and thus on its micro F-measure. The detailed experimental results for Task 4 over training data can be found in the following links:

Table 3: Results for Task 1 of the QALD challenge. Table extracted from [27].

Test	**WDAqua**	**ganswer2**	**AMAL**
Language	**en**	**en**	**fr**
Error count		3	
Micro Precision	0.080	0.322	0.998
Micro Recall	0.006	0.127	0.989
Micro F1-measure	0.012	0.182	0.993
Macro Precision	0.162	0.487	0.720
Macro Recall	0.160	0.498	0.720
Macro F1-measure	0.143	0.469	0.720
Train	**WDAqua**	**ganswer2**	**AMAL**
Language	**en**	**en**	**fr**
Error count			
Micro Precision	-	0.113	0.971
Micro Recall	-	0.561	0.697
Micro F1-measure	-	0.189	0.811
Macro Precision	0.490	0.557	0.750
Macro Recall	0.540	0.592	0.751
Macro F1-measure	0.510	0.556	0.751

- WDAqua: http://master.project-hobbit.eu/#/experiments/details?id=1498647883035
- Sorokin and Gurevych: http://master.project-hobbit.eu/#/experiments/details?id=1498647941734

For the testing data they can be found in the following links:

- WDAqua: http://master.project-hobbit.eu/#/experiments/details?id=1498647794373
- Sorokin and Gurevych: http://master.project-hobbit.eu/#/experiments/details?id=1498647917506

4.5 Conclusions

The successful organization of QALD in 2017 by HOBBIT will serve as the basis for the future editions of the challenge. We aim at further simplifying the participation process and offering leaderboards so that the challenge participants can easily preview their systems' performance. Moreover,

Table 4: Results for Task 4 of the QALD challenge. Table extracted from [27].

Test dataset	**WDAqua**	**Sorokin and Gurevych**
Micro Precision	0.392	0.428
Micro Recall	0.082	0.030
Micro F1-measure	**0.136**	0.057
Macro Precision	0.739	0.661
Macro Recall	0.606	0.430
Macro F1-measure	**0.552**	0.427
Train dataset	**WDAqua**	**Sorokin and Gurevych**
Micro Precision	0.172	0.295
Micro Recall	0.112	0.070
Micro F1-measure	**0.136**	0.113
Macro Precision	0.759	0.774
Macro Recall	0.710	0.756
Macro F1-measure	0.636	**0.645**

given the feedback from the participating teams, we will add new key performance indicators to also account for the capability of a system to know which questions it cannot answer and take confidence scores for answers into account. Also, the HOBBIT platform and its documentation will be improved to encourage more teams to participate.

More details on the QALD challenge, systems and results can be found on the papers mentioned in Section 4.1.

5 OKE Challenge Results Overview

5.1 Definition of Tasks

The Open Knowledge Extraction (OKE) challenge has the ambition to provide a reference framework for research on Knowledge Extraction from text for the semantic web by re-defining a number of tasks (typically from information and knowledge extraction), taking into account specific semantic web requirements. It thus invites researchers and practitioners from academia as well as industry to compete to the aim of pushing further the state of the art of knowledge extraction from text for the semantic web. OKE in 2017 focused on the following three tasks:

- Task 1: Focused Named Entity Identification and Linking
- Task 2: Broader Named Entity Identification and Linking

Table 5: Types and subtype and instance examples for Task 2 of the OKE challenge. Table extracted from [25].

Type	Subtypes	Instances
Activity	Game, Sport	Baseball,Chess
Agent	Organisation, Person	Leipzig_University
Award	Decoration, NobelPrize	Humanitas_Prize
Disease		Diabetes_mellitus
EthnicGroup		Javanese_people
Event	Competition, PersonalEvent	Battle_of_Leipzig
Language	ProgrammingLanguage	English_language
MeanOfTransportation	Aircraft, Train	Airbus_A300
PersonFunction	PoliticalFunction	PoliticalFunction
Place	Monument, WineRegion	Beaujolais, Leipzig
Species	Animal, Bacteria	Cat, Cucumibacter
Work	Artwork, Film	Actrius, Debian

- Task 3: Focused Musical Named Entity Recognition and Linking
- Task 4: Knowledge Extraction

In more detail, *Task 1* aims at the identification and linking of entities of a given, limited set of entity types. It is a two-step process, including the identification of named entities (**Recognition step**) and the linking of those entities to resources in DBpedia (**D2KB step**). The task is limited to a subset of three DBpedia ontology types; Person, Place and Organisation. *Task 2* extends Task 1 to more DBpedia ontology types. Besides the three types mentioned above, a competing system might have to identify other types of entities and to link these entities as well. Table 5 provides a complete list of types that are considered in this task. Example subtypes of the corresponding class, if they exist, as well as example instances are also shown. *Task 3* consists of two subtasks; (a) focused musical named entity identification and classification and (b) linking to the MBL knowledge base that is based on MusicBrainz. The first subtask consists of the identification (**Recognition step**) and classification (**Typing step**) of named entities. The task is limited to a subset of three MBL ontology types; Artist, Album and Song. For the second subtask, the entities recognized in the first subtask must be linked to the corresponding resources in MBL if existing or to generate a URI for the emerging entity (**D2KB step**). A system has to fulfill both subtasks in order to participate in Task 3. *Task 4* aims at extracting knowledge from a given text and to formalize the knowledge in RDF triples. DBpedia is considered as the knowledge base in this task.

OKE was organized in conjunction with the ESWC 2017 conference, where systems participating in the challenge and their results were presented to the public in a dedicated workshop session. The papers describing the systems submitted to the OKE challenge were peer-reviewed by experts and the most promising systems were invited to participate in the challenge. The OKE challenge papers

have been published by Springer on the proceedings volume *Dragoni M., Solanki M. and Blomqvist E. (eds), Semantic Web Challenges, Communications in Computer and Information Science, vol. 769, 2017*[22]. In particular, in [25] the challenge organizers presented an overview of the challenge results and baseline systems, while in [14] the challenge participants described their system.

5.2 Participating Systems

The challenge attracted three teams/systems. One team was rejected following the reviewers' comments, while another team, although it was accepted, withdrew from the challenge a few days before the ESWC 2017 conference. The remaining one system and the baseline system provided by the challenge organizers participated in the challenge in Tasks 1, 2 and 3. No systems were submitted for Task 4.

ADEL [14] is an adaptive entity recognition and linking framework based on a hybrid approach that combines various extraction methods to improve the recognition level and an efficient knowledge base indexing process to increase the efficiency of the linking step. It deals with fine-grained entity types, either generic or domain specific. It also can flexibly disambiguate entities from different knowledge bases.

FOX [24] serves as the baseline system in the OKE challenge and has been introduced in 2014 as an ensemble learning-based approach combining several diverse state of the art named entity recognition approaches and is based on the work in [13]. The FOX framework outperforms the current state of the art entity recognizers. It relies on AGDISTIS [28] to perform named entity disambiguation. AGDISTIS is a pure entity linking approach (D2KB) based on string similarity measures, an expansion heuristic for labels to cope with co-referencing and the graph-based HITS algorithm.

5.3 Evaluation Metrics

The systems participating in the challenge were evaluated using recall, precision, F1-measure and β. The equations below formalize these performance measures, where true positives are denoted as TP_d, false positives as FP_d and false negatives as FN_d. The performance measures below are defined on a per document d basis, on which named entities should be identified. Each challenge task consists of several documents and we thus micro averaged the performance over all the documents to calculate the overall performance of a system.

$$precision_d = \frac{TP_d}{TP_d + FP_d}$$

$$recall_d = \frac{TP_d}{TP_d + FN_d}$$

$$f1_d = 2 \cdot \frac{precision_d \cdot recall_d}{precision_d + recall_d}$$

Let D be a set of documents for which β should be calculated and let t_d be the time (in seconds) a system needs for the annotation of a document d. Then the β value is the amount of F1-measure points a system achieves per second for a given amount of documents.

[22] https://doi.org/10.1007/978-3-319-69146-6

$$\beta = \frac{\sum_{d \in D} f1_d}{\sum_{d \in D} t_d}$$

For matching the entity annotation positions identified by a system and the correct entity markings of the documents we used the *weak annotation matching* defined in [29]. Hence, an entity is counted as having the correct position, if its position overlaps with the correct position of the entity inside the document.

5.4 Results & Achievements

5.4.1 Benchmarking Scenarios

The benchmarking suite for named entity recognition and linking implemented within the HOBBIT platform reuses some of the concepts developed within the open-source project Gerbil [29]. These concepts were migrated and adapted to the HOBBIT architecture. The HOBBIT platform provides two different benchmarking implementations described in the following subsections, which were used to evaluate the participating systems.

Scenario A: Quality-focused benchmarking

The first type of benchmarking focuses on the measurement of quality a system achieves on a given set of documents. We assume that each task consists of a set of documents. The documents are sent to the benchmarked system one at a time. The benchmarked system generates a response and sends it back before receiving the next document. That means that the benchmarked system can be configured to concentrate all its resources on a single request and does not need to scale to a large number of requests. In this benchmarking scenario we rely on manually created gold standard datasets. The goal in this scenario is to achieve a high F1-measure in a quality-focused benchmarking.

Scenario B: Performance-focused benchmarking

The second benchmarking approach aims to put a high load on the benchmarked system to evaluate its runtime and quality in terms of precision, recall and F1-measure. This approach, hence, focuses on the ability of a system to annotate documents in parallel with an increasing amount of load. A large amount of synthetic documents is created using Bengal [12][23]. These documents are sent to the system in parallel without waiting for responses for previous requests, but with predefined delays between the single documents. During a first phase of the benchmarking, the generated workload equals 1 document per second. After the 80 documents of this first phase have been sent, the next phase is started using half of the delay of the previous time. This is done for 6 phases. In the seventh and last phase all 80 documents of the phase are sent without a delay, this leads to workloads of $\{1, 2, 4, 8, 16, 32, 80\}$ documents per second during the different phases. The performance of a system is measured by β which is defined in Section 5.3. The scenarios goal is to achieve a high β value.

5.4.2 Task Results

In this section the results of the participating systems are presented. Tables 6 and 7 depict the results for Tasks 1 and 2, while Tables 8 and 9 those for Tasks 3A and 3B. The tables show the overall performance in terms of precision, recall and F1-measure in the first three rows. The last two rows in each table report on the averaged time in seconds a system needs to process a document and the errors

[23] http://github.com/aksw/bengal

Table 6: Results for Task 1 of the OKE challenge. Table extracted from [25].

Experiment Type	Micro measures	Scenario A		Scenario B	
		ADEL	FOX	ADEL	FOX
A2KB	Precision	33.24	53.61	18.28	59.12
	Recall	30.18	46.72	22.36	72.51
	F1-measure	31.64	**49.93**	20.12	**65.15**
Recognition	Precision	91.62	92.47	74.39	73.27
	Recall	83.20	80.58	90.98	89.85
	F1-measure	**87.21**	86.12	**81.85**	80.72
D2KB	Precision	40.15	61.96	28.03	93.87
	Recall	27.82	41.47	19.26	66.99
	F1-measure	32.87	**49.69**	22.83	**78.19**
	Time	7.98	6.98	231.31	179.29
	Errors	0	0	6	1

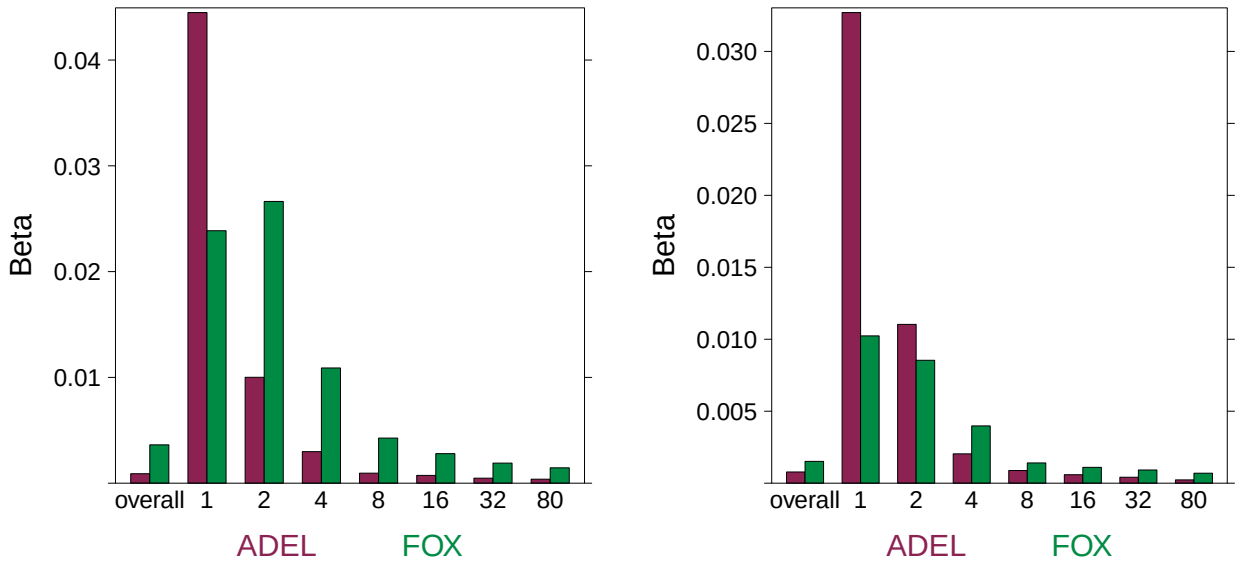
a system triggers (i.e. the number of documents that cause an error to the system). Further Tables 6 to 8 show the interim results for the Recognition step in the next three rows and for the D2KB or Typing step, depending on the underlying task, in the following three rows. For Task 3B there are no interim results since there are no interim steps in this subtask.

Task 1 results

The measured values for scenario A in Table 6 show that ADEL outperforms FOX slightly with +1.09% F1-measure in the Recognition step. In the D2KB step, FOX outperforms ADEL clearly with +16.82% F1-measure. Overall, FOX outperforms ADEL with +18.29% in Task 1 in scenario A.

In scenario B, the results are similar to scenario A. In the Recognition step ADEL outperforms FOX slightly as well as FOX outperforms ADEL clearly in the D2KB step. Overall, FOX reaches the highest value in scenario B with 65.15% F1-measure while ADEL reaches 20.12% F1-measure. With 6 and 1 errors, the error rates of ADEL and FOX are low compared to the number of 560 documents they had to annotate in this scenario.

Figure 12 depicts on the left side the detailed results in terms of the β measure for Task 1 in scenario B. Surprisingly, ADEL reaches a clearly higher β value than FOX in the first phase for one document request per second. This is caused by the fast runtime of ADEL compensating its lower F1-score during that phase. In the following phases, the runtime of both systems increases, a clear sign that they are receiving requests to annotate document while they are still working on other documents. However, compared to FOX, the time that ADEL needs per document increases much more. Since the F1-score of both systems are similar over all phases, but the time needed per document of FOX does not increase as much as it does for ADEL the β value of FOX remains higher than the value for ADEL. The observation of the increasing of processing time can be also seen in the comparison of the



(a) β on Task 1.

(b) β on Task 2.

Figure 12: β values on several numbers of requests and overall for Tasks 1 and 2 of the OKE challenge. Figure extracted from [25].

overall values of scenario A and B. While ADEL needs 14% more time per document on average in scenario A, this increases to 29% in scenario B. Together with the higher F1-score, the lower runtime of FOX leads to an overall β value which is four times higher than the value of ADEL.

Task 2 results

The measured values for scenario A in Table 7 show that ADEL outperforms FOX slightly with +4.83% F1-measure in the Recognition step. In the D2KB step, FOX outperforms ADEL clearly with +14.02% F1-measure. Overall, FOX outperforms ADEL with +16.02% in Task 2 in scenario A. Contrary to Task 1, ADEL is nearly twice as fast as FOX in scenario A.

In scenario B, the results are similar to scenario A. In the Recognition step ADEL outperforms FOX as well as FOX outperforms ADEL clearly in the D2KB step. Overall, FOX reaches the highest value in scenario B with 42.22% F1-measure while ADEL reaches 18.15% F1-measure.

Figure 12 depicts on the right side the detailed results in terms of the β measure for Task 2 in scenario B. Similar to Task 1, ADEL reaches a clearly higher β value than FOX in the first two phases. This is again caused by the lower runtime of ADEL that compensates its lower F1-score. In all other phases FOX reaches a higher β value because, as in Task 1, the runtime of ADEL increases much more than the runtime of FOX when it receives many requests in a short amount of time. Overall, FOX nearly reaches a β value twice as high as the value achieved by ADEL. It is also worth noting that this is the only experiment, in which the error rate of one of the systems is increased. For 57 of the 560 documents, ADEL responded with an error code. Nearly all of these errors occurred during the last three phases (9, 26 and 21 respectively). Since the documents are chosen randomly and ADEL reported nearly no errors in the phases before, it is possible that they are related to the high load that ADEL receives during these phases.

Task 3 results

Table 7: Results for Task 2 of the OKE challenge. Table extracted from [25].

Experiment Type	Micro measures	Scenario A		Scenario B	
		ADEL	FOX	ADEL	FOX
A2KB	Precision	31.40	56.15	17.44	44.90
	Recall	28.14	38.53	18.93	39.83
	F1-measure	29.68	**45.70**	18.15	**42.22**
Recognition	Precision	87.68	95.90	72.31	74.64
	Recall	78.57	65.80	78.50	66.21
	F1-measure	**82.88**	78.05	**75.27**	70.17
D2KB	Precision	39.93	63.42	28.57	82.38
	Recall	25.76	35.28	17.47	36.92
	F1-measure	31.32	**45.34**	21.68	**51.00**
	Time	4.60	7.66	261.48	245.99
	Errors	0	1	57	0

The measured values for Task 3A are depicted in Table 8. FOX reaches a higher F1-measure than ADEL, 55.27% to 47.66% in the Recognition step. In the Typing step ADEL reaches a higher F1-measure, since FOX is not supporting this subtask due to the lack of the support of the music entity types. Overall, ADEL reaches the highest value with 27.12% F1-measure on this task.

The measured values for Task 3B are depicted in Table 9. Both systems, ADEL and FOX, attain low performance in this task. ADEL achieves 5.83% and FOX a slightly higher value with 6.66%. It is noteworthy that FOX processed the documents faster with 9.15s/doc in this subtask than ADEL with 36.96s/doc. Additionally, FOX encountered no errors in comparison to ADEL for which 16 errors have been reported.

5.5 Conclusions

The winner of Tasks 1 and 2 in both scenarios A and B is the baseline system FOX. For Task 3A the winner is ADEL, since FOX is not supporting all subtasks. For Task 3B the winner is FOX again. Since the advantage ADEL has in Task 3A is larger than the difference between FOX and ADEL in Task 3B, ADEL is the overall winner of Task 3.

The results on Task 1 and 2 suggest, that the Recognition component in ADEL achieved a higher F-measure than the respective component in FOX, but its linking component showed a worse performance than the respective component in FOX. Thus, it would be interesting to investigate the performance of the composition of the Recognition component of ADEL together with the linking component in FOX in these tasks. The results on Task 3 in the music domain suggest that the Recognition component of FOX achieved a better F-measure than ADEL. FOX is not supporting the music entity types in

its current version, thus it would be interesting to investigate the performance of an extended version that supports these types compared to ADEL in this task.

Table 8: Results for Task 3A of the OKE challenge. Table extracted from [25].

Experiment Type	Micro measures	ADEL	FOX
RT2KB	Precision	26.99	0
	Recall	27.24	0
	F1-measure	**27.12**	0
Recognition	Precision	35.03	63.02
	Recall	74.57	49.21
	F1-measure	47.66	**55.27**
Typing	Precision	64.33	0
	Recall	64.91	0
	F1-measure	**64.62**	0
	Time	37.19	7.82
	Errors	16	0

Table 9: Results for Task 3B of the OKE challenge. Table extracted from [25].

Experiment Type	Micro measures	ADEL	FOX
D2KB	Precision	6.82	10.10
	Recall	5.10	4.97
	F1-measure	5.83	**6.66**
	Time	36.96	9.15
	Errors	16	0

The successful organization of OKE in 2017 by HOBBIT will serve as the basis for the future editions of the challenge. We aim at further simplifying the participation process and offering leaderboards so that the challenge participants can easily preview their systems' performance. Moreover, given the feedback from the participating teams, the HOBBIT platform and its documentation will be improved to encourage more teams to participate.

More details on the OKE challenge, systems and results can be found on the papers mentioned in Section 5.1.

6 Link Discovery Track Results Overview

6.1 Definition of Tasks

The "Link Discovery Track" was accepted at the OAEI OM 2017 Workshop at ISWC 2017. OM workshop conducted an extensive and rigorous evaluation of ontology matching and instance matching (link discovery) approaches through the OAEI (Ontology Alignment Evaluation Initiative) 2017 campaign.

In the Link Discovery Track two benchmark generators were proposed to deal with *link discovery* for spatial data represented as *trajectories* i.e., sequences of longitude, latitude pairs. The Link Discovery track employs the HOBBIT platform[24].

The Link Discovery Track aimed to test the performance of Link Discovery tools that implement string-based as well as topological approaches for identifying matching spatial entities. The different frameworks have been evaluated for both accuracy (precision, recall and F-measure) and time performance.

In the Link Discovery Task TomTom [25] Data have been employed for the creation of the appropriate benchmarks. TomTom datasets contain representations of traces (GPS fixes). Each trace comprises a number of points. Each point has time stamp, longitude, latitude and speed (value and metric). The points are sorted by time stamp of the corresponding GPS fix (ascending). Each task of the HOBBIT Link Discovery Track consists of two datasets with different number of instances to match, namely the Sandbox and the Mainbox.

All in all the Link Discovery Track comprises the following tasks:

- *Task 1 (Linking)* measures how well the systems can match traces that have been modified using string-based approaches along with addition and deletion of intermediate points. Since TomTom datasets only contain coordinates, in order to apply string-based modifications implemented in LANCE [19] we have replaced a number of those points with labels retrieved from Linked Data spatial datasets using the Google Maps[26], Foursquare[27] and Nominatim Openstreetmap[28] APIs. This task also contains modifications on date and coordinate formats.
- *Task 2 (Spatial)* measures how well the systems can identify the DE-9IM (Dimensionally Extended nine-Intersection Model) topological relations. The supported spatial relations are the following: *Equals, Disjoint, Touches, Contains/Within, Covers/CoveredBy, Intersects, Crosses, Overlaps*. The traces are represented in the Well-known text (WKT) format. For each relation, a different pair of source and target datasets is given to the participants.

6.2 Participating Systems

The participating systems of the Linking task are two: AgreementMakerLight (AML) and OntoIdea systems. Both systems have been elected to publish results papers on the OM website [1,5].

In essence AgreementMakerLight (AML) is an automated ontology matching system. OntoIdea is an idea matching system. In the context of the idea generation process, each idea is represented by a set

[24] https://project-hobbit.eu/outcomes/hobbit-platform/
[25] https://www.tomtom.com/
[26] https://developers.google.com/maps/
[27] https://developer.foursquare.com/
[28] http://nominatim.openstreetmap.org/

of instances from DBpedia describing the main concepts of the idea. Then, the developed matching system are applied to compute the similarity between a set of instances that represent the ideas.

Furthermore, the participating systems to the Spatial task were four: AgreementMakerLight (AML), OntoIdea, Rapid Discovery of Topological Relations (RADON) and Silk systems. RADON and Silk systems had already been described in [20] and [21].

In essence RADON performs the discovery of topological relations between geospatial resources according to the DE9-IM standard. Furthermore, Silk is a framework for Spatial and Temporal Link Discovery.

6.3 Results & Achievements

6.3.1 Task 1 (Linking)

Regarding Task 1 an instance in the source dataset has one matching counterpart in the target dataset. For the *Linking Task*, the Sandbox scale is 100 instances while the Mainbox scale is 5K instances. The participants were asked to match traces in the source and the target datasets. For evaluation, a ground truth was built containing the set of expected links where an instance i_1 in the source dataset is associated with an instance j_1 in the target dataset that has been generated as an altered description of i_1. The way that the transformations were done, was to apply value-based, and structure-based transformations on different triples pertaining to instances of class Trace.

Table 10: HOBBIT Link Discovery Linking Task (Sandbox)

Sandbox task

	Precision	**Recall**	**F-measure**	**Run Time**
AML	1.000	1.000	1.000	11722
OntoIdea	0.990	0.990	0.990	19806

Table 11: HOBBIT Link Discovery Linking Task (Mainbox)

Mainbox task

	Precision	**Recall**	**F-measure**	**Run Time**
AML	1.000	1.000	1.000	134456
OntoIdea	Platform Time Limit (75 mins)			

The systems were judged on the basis of *precision*, *recall*, *F-measure* and *runtime* results that are shown in Tables 10 and 11. Both AML and OntoIdea systems return high precision and recall capturing all the correct links. As far as runtime is concerned, for the Sandbox dataset, AML needed less time than OntoIdea and for the Mainbox dataset, AML completes the task with perfect results in contrast to OntoIdea that was not able to complete it and stopped when it hit the platform time limit (75 mins). Datasets, reference alignments, and task results are available on the HOBBIT website: https://project-hobbit.eu/challenges/om2017/.

6.3.2 Task 2 (Spatial)

Regarding Task 2 given a LineString source geometry s, a LineString target geometry t and a DE-9IM topological relation r, the participants were asked to match an instance from s with one or more instances in t such as their Intersection Matrix follows the definition of r. For evaluation, a ground truth was built using RADON [20] containing the set of expected links where an instance i_1 in the source dataset is associated with one or more instances in the target dataset that has been generated as an altered description of i_1. For the *Spatial Task*, the Sandbox scale is 10 instances and the Mainbox scale is 2K instances.

The systems were judged on the basis of *precision*, *recall*, *F-measure* and *runtime* results shown in Table 12 and Figures 13 and 14. Only time performance is presented. Precision, recall and F-measure are not presented as all were equal to 1.0 except *OntoIdea* that reports for the *Touches* and *Overlaps* relations value 0.99. Furthermore, Silk is not participating in relations *Covers* and *Covered By* and OntoIdea is not participating in relation *Disjoint*.

From the results we it is detected that:

- **OntoIdea** has the best performance in the Sandbox dataset *while* in the Mainbox dataset the runtime increases and the system seems to not be able to handle large datasets easily.
- **Silk** shows a similar behaviour as **OntoIdea**.
- **RADON** and **AML** systems appear to handle the growth of the dataset size smoother.
- **AML** does not provide any results for the *Disjoint* relation since it reaches the platform time limit

Datasets, reference alignments, and task results are available on the HOBBIT website: https://project-hobbit.eu/challenges/om2017/.

Relation	Systems	Sandbox Run Time	Mainbox Run Time
EQUALS	AML	8157	10284
	OntoIdea	1531	567169
	RADON	2215	4680
	Silk	4059	125967
DISJOINT	AML	7173	Platform Time Limit (75 mins)
	OntoIdea	Not participating	
	RADON	1558	19214
	Silk	3224	257877
TOUCHES	AML	11207	20252
	OntoIdea	4712	473430
	RADON	2672	485765
	Silk	4805	1777747
CONTAINS	AML	9191	16966
	OntoIdea	1489	223857
	RADON	2228	6937
	Silk	4160	83958
WITHIN	AML	10186	12308
	OntoIdea	4517	236506
	RADON	2203	5036
	Silk	4037	88758
COVERS	AML	7177	11859
	OntoIdea	1503	313298
	RADON	2180	6772
	Silk	Not participating	
COVERED BY	AML	8184	14703
	OntoIdea	1467	304509
	RADON	2132	4721
	Silk	Not participating	
INTERSECTS	AML	9269	66681
	OntoIdea	1505	510938
	RADON	2737	339742
	Silk	3582	1718035
CROSSES	AML	8224	19385
	OntoIdea	1509	461693
	RADON	2131	8490
	Silk	3917	203763
OVERLAPS	AML	10223	194838
	OntoIdea	1486	530752
	RADON	2167	60801
	Silk	4217	464382

Table 12: Spatial Benchmark Results.

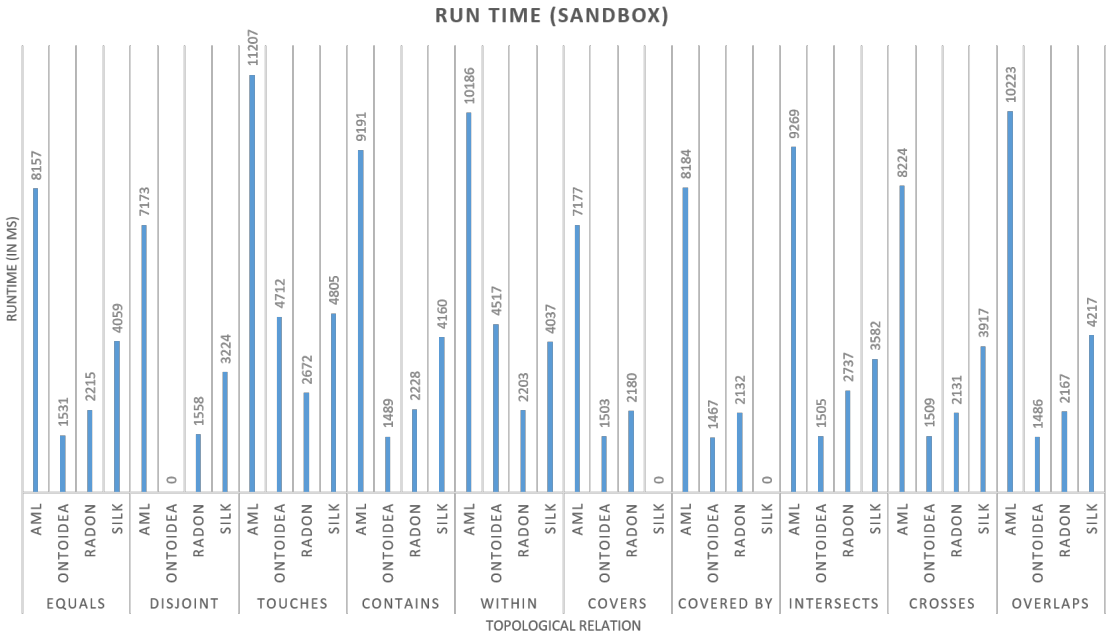


Figure 13: HOBBIT Link Discovery Spatial Task (Sandbox)

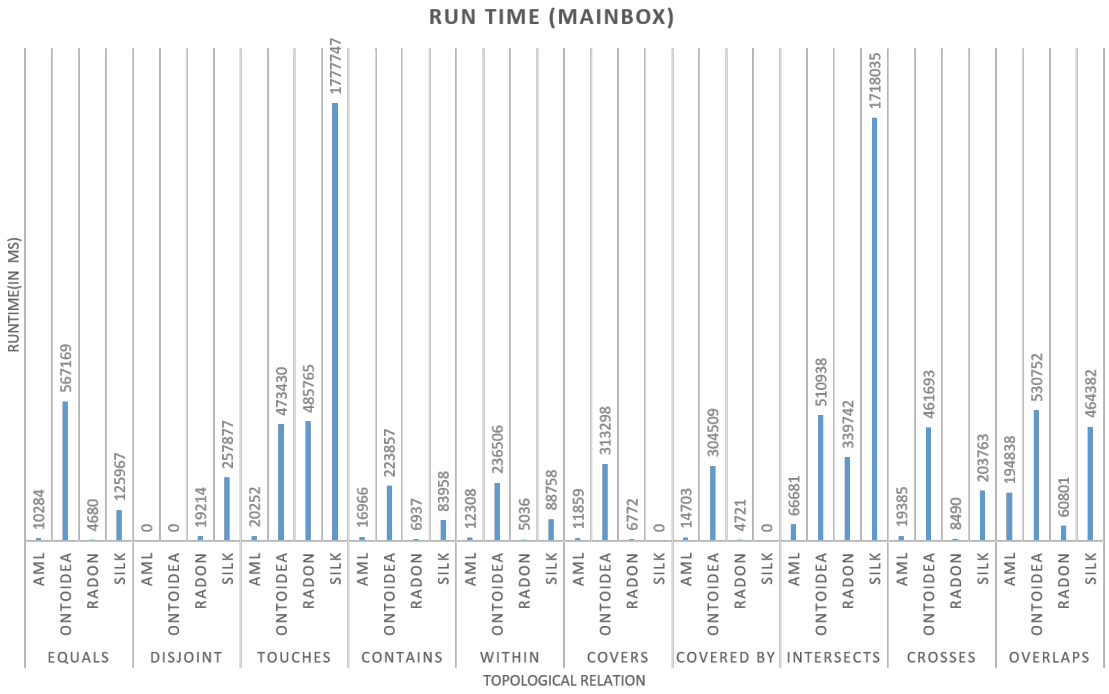


Figure 14: HOBBIT Link Discovery Spatial Task (Mainbox)

7 DEBS Grand Challenge Results Overview

7.1 Definition of Tasks

The DEBS Grand Challenge is a series of annual challenges joint to the annual ACM International Conference on Distributed and Event-Based Systems (DEBS) which targets the evaluation of event-based systems for real-time analytics in different domains. The 2017 DEBS Grand Challenge[29] was focused on the task of analyzing RDF streaming data generated by digital and analogue sensors embedded within manufacturing equipment. The analysis aims at detecting anomalies in the behavior of manufacturing equipment using machine learning-based classification.

In more detail, two scenarios that relate to the problem of automatic detection of anomalies for manufacturing equipment were considered. The overall goal of both scenarios was to detect abnormal behavior of a manufacturing machine based on the observation of the *stream* of measurements provided by such a machine. Participating systems were required to cluster the data produced by each sensor and model the state transitions between the observed clusters as a Markov chain. Based on this classification, anomalies were detected as sequences of transitions that happen with a probability lower than a given threshold. The difference between the two scenarios is that in the *first scenario* the number of machines to observe is fixed, while in the *second scenario* new machines dynamically join and leave the set of observable machines. Two types of machines were considered; (a) injection molding machines and (b) assembly machines. Injection molding machines are equipped with sensors that measure various parameters of the production process: distance, pressure, time, frequency, volume, temperature, time, speed and force. Assembly machines are equipped with three energy meters.

The 2017 DEBS Grand Challenge has been co-organized by the HOBBIT project that provided the dataset for the challenge, as well as the HOBBIT platform for the evaluation of the submitted systems. The challenge papers have been published by ACM as part of the DEBS 2017 conference proceedings volume, *DEBS '17: Proceedings of the 11th ACM International Conference on Distributed and Event-based Systems, ACM, New York, NY, USA, 2017*[30]. Specifically, in [8] the challenge organizers presented an overview of the challenge, while in [2, 3, 4, 9, 10, 17, 30] the challenge participants described their systems.

7.2 Participating Systems

Seven teams/systems participated in DEBS Grand Challenge, addressing both scenarios.

The team from the **WSO2 Company, USA** [2] built their solution on top of Siddhi, the open source complex event processing engine of WSO2 DAS. As mentioned in their paper the Siddhi Query Language associated with WSO2 DAS was used to formulate the challenge query for the solution. The Disruptor ring buffer was applied to organize concurrent processing between threads. All the threads were divided in the following functionalities: RabbitMQ pool, Data Extraction pool, Sorter Thread, Disruptor Thread and Siddhi Handlers.

The winning team from the **Alexandru Ioan Cuza University, Romania** [3] developed their solution in the Java language, using a custom pipelined architecture. As mentioned in their paper the reason for choosing Java was based mostly on the fact that the testing platform was written in Java, involving several standard wrapper classes facilitating the integration of the solution with

[29] https://project-hobbit.eu/challenges/debs-grand-challenge/ and http://lsd1.ls.fi.upm.es/debs2017/call-for-grand-challenge-solutions/

[30] https://dl.acm.org/citation.cfm?id=3093742

the benchmark. The rationale of preferring a custom architecture to a dedicated stream processing platform (e.g., Storm) had mainly two grounds. The first was due to some optimization cases which are selectively triggered based on the window values configuration and which involve all processing stages, making improper to adhere to the specific operator-like separation that stream processing platforms typically imply. The second ground was the single node approach that was adopted in the end.

The team from **Chungnam National University, Korea** [4] created a custom, highly parallel solution with internal queues between operators. As mentioned in their paper to save the tuples order a sequence-tagged event method was applied, which sorts the results of parallel processing according to numbers by which events have been tagged before processing. The performance of the solution was tested by the team on 50K events, varying window sizes and queue modes (no queue, single queue, multiple queues).

The team from the **Rice University, USA** [9] presented two implementations of their solutions using Java and C++ programming languages. As mentioned in their paper, they implemented their customized multi-threaded RDF parser in Java and parallel anomaly detector using OpenMP. Local performance assessments on different window sizes were made by sending 10,000 data points which demonstrated a dependency between the overall latency and the number of parallel threads.

The team from **University of Stuttgart, Germany** [10], which won the audience award for the most elegant solution, presented the StreamLearner, a custom solution which provides event-based stream processing and powerful machine learning functionality. Experiments performed by the team and reported in their paper demonstrate the scalability of the StreamLearner architecture and a throughput of up to 500 events per second using the proposed algorithms for incremental machine learning model updates.

The team from **Israel Institute of Technology** [17] considered Apache Storm, Apache Spark and Apache Flink as potential candidates for their solution. As mentioned in their paper they finally picked Apache Flink due to its: (i) documented higher throughput and lower latency, (ii) API at high to low abstraction level, (iii) native time-based window and out-of-order managing mechanisms based on event-time, (iv) streamlined performance tuning, (v) API and engine being coded with HOBBIT's reference language (Java).

The team from **Insight Centre for Data Analytics, Ireland** [30] also examined several open-source streaming frameworks (Apache Storm, Apache Spark and Apache Flink) as candidates for their solution. As mentioned in their paper they decided not to use Storm because it does not support batching capabilities. Between Spark and Flink they elected to use Flink, even though Spark is more mature, because it processes tasks with lower latency due to its pipelined execution.

7.3 Results & Achievements

To evaluate the systems submitted to the challenge, three experimental cases were considered. Each case was defined by a number of fixed and volatile parameters. The volatile parameters were the amount of machines and the run mode, which can be either static or dynamic. In static mode experiments the number of machines considered remains the same for the entire experiment, while in dynamic mode experiments the amount of machines is incremented by staring from an initial fixed value and dynamically adding new machines after every N sensor measurements (N is a fixed parameter). Specifically, the first case concerned static mode with 1 machine, the second static mode with 10 machines and the third dynamic mode with 10 machines initially and a new machine joining every $N = 1$ measurements.

Systems were ranked based on the latency of the provided solution over the three experimental

Table 13: Results for the first experimental case of the DEBS Grand Challenge (static mode, 1 machine).

Team	Throughput (bytes/s)	Latency (ms)
Alexandru Ioan Cuza University	1,682,096	39,194
Rice University (Java)	2,332,643	48,043
Rice University (C++)	2,359,601	49,879
Chungnam National University	1,597,444	80,124
University of Stuttgart	1,661,759	99,540
Israel Institute of Technology	2,261,829	261,610
Insight Centre for Data Analytics	2,046,777	328,281
WSO2	1,469,544	610,343

cases. Tables 13 to 15 present the attained latency values of the systems in ascending order. Missing latency means that the system did not found all anomalies correctly. Also the throughput of each system is given. The results clearly show that the solution developed by the Alexandru Ioan Cuza University team has the lowest latency.

7.4 Conclusions

The successful organization of the 2017 DEBS Grand Challenge by HOBBIT will serve as the basis for the future editions of the challenge. We aim at further simplifying the participation process and offering leaderboards so that the challenge participants can easily preview their systems' performance. Moreover, given the feedback from the participating teams, the HOBBIT platform and its documentation will be improved to encourage more teams to participate. More details on the DEBS Grand

Table 14: Results for the second experimental case of the DEBS Grand Challenge (static mode, 10 machines).

Team	Throughput (bytes/s)	Latency (ms)
Alexandru Ioan Cuza University	1,526,773	39,760
Rice University (C++)	1,499,941	49,928
Rice University (Java)	1,483,409	64,232
University of Stuttgart	1,525,247	96,542
Chungnam National University	2,219,931	214,688
Israel Institute of Technology	1,487,197	—
Insight Centre for Data Analytics	1,411,077	533,224
WSO2	1,428,495	1,791,852

Table 15: Results for the third experimental case of the DEBS Grand Challenge (dynamic mode, 1 to 10 machines, new machine joining every $N = 1$ measurements).

Team	**Throughput (bytes/s)**	**Latency (ms)**
Alexandru Ioan Cuza University	1,561,631	38,344
Rice University (C++)	2,219,111	50,704
Rice University (Java)	1,521,045	66,664
WSO2	1,424,234	98,959
Chungnam National University	1,527,820	421,856
University of Stuttgart	1,495,759	1,393,908
Israel Institute of Technology	1,487,190	—
Insight Centre for Data Analytics	1,413,646	1,907,376

Challenge, systems and results can be found on the papers mentioned in Section 7.1.

8 Conclusions

The HOBBIT project has so far successfully organized five challenges. The MOCHA, OKE and QALD challenges were organized in conjunction with the ESWC 2017 conference. Also, HOBBIT was responsible for the 2017 DEBS Grand Challenge that annually runs as part of the DEBS conference, as well as the Link Discovery Task at the 2017 OAEI campaign which was held under the Ontology Matching workshop at the ISWC 2017 conference. Participating systems were evaluated using the 8 HOBBIT benchmarks and platform. The results suggest that a fair amount of work still needs to be carried out by the community to reach the necessary scalability. The challenges were the first outing of the benchmarking platform, and the FAIR evaluation of Big (Linked) Data processing technologies was received very positively by the community.

The activities foreseen for the near future include the completion of the second version of the benchmarks and mimicking algorithms, and the organization of the second round of HOBBIT challenges. Top priority will be to address the participants' feedback from the first round.

In particular, we identified that challenge participants were facing difficulties in integrating their systems into the HOBBIT platform. In the second version of the HOBBIT platform (that is currently under development) we foresee the simplification of the procedures and technologies needed to integrate and execute a system on the platform. We have also improved the platform's documentation and corresponding instructions provided on the challenges' websites on how to submit and test systems.

Already, preparations for the organization of the second series of the challenges have started. HOBBIT will organize the MOCHA, OKE and SQA (Scalable Question Answering - an offspring of QALD), challenges at the ESWC 2018 conference. It will also contribute the Link Discovery task in the OAEI 2017.5 challenge which will run at ESWC 2018. HOBBIT will again be responsible for the DEBS Grand Challenge in 2018 and plans to run the Link Discovery task as part of the 2018 OAEI campaign at ISWC 2018. Finally, HOBBIT has already launched the MOCHA, OKE, SQA and StreaML open challenges[31].

[31] https://project-hobbit.eu/open-challenges/

References

[1] Maximilian Mackeprang Abderrahmane Khiat. I-Match and OntoIdea Results for OAEI 2017. http://www.dit.unitn.it/~pavel/om2017/papers/oaei17_paper4.pdf, 2017.

[2] Nihla Akram, Sachini Siriwardene, Malith Jayasinghe, Miyuru Dayarathna, Isuru Perera, Seshika Fernando, Srinath Perera, Upul Bandara, and Sriskandarajah Suhothayan. Anomaly detection of manufacturing equipment via high performance rdf data stream processing: Grand challenge. In *Proceedings of the 11th ACM International Conference on Distributed and Event-based Systems*, DEBS '17, pages 280–285, New York, NY, USA, 2017. ACM.

[3] Ciprian Amariei, Paul Diac, and Emanuel Onica. Optimized stage processing for anomaly detection on numerical data streams: Grand challenge. In *Proceedings of the 11th ACM International Conference on Distributed and Event-based Systems*, DEBS '17, pages 286–291, New York, NY, USA, 2017. ACM.

[4] Joong-Hyun Choi, Kang-Woo Lee, Hyungkun Jung, and Eun-Sun Cho. Runtime anomaly detection method in smart factories using machine learning on rdf event streams: Grand challenge. In *Proceedings of the 11th ACM International Conference on Distributed and Event-based Systems*, DEBS '17, pages 304–309, New York, NY, USA, 2017. ACM.

[5] Vivek Shivaprabhu Isabela Mott Catia Pesquita Francisco Couto Isabel Cruz Daniel Faria, Booma S. Balasubramani. Results of AML in OAEI 2017. http://www.dit.unitn.it/~pavel/om2017/papers/oaei17_paper2.pdf, 2017.

[6] Dennis Diefenbach, Kamal Singh, and Pierre Maret. *WDAqua-core0: A Question Answering Component for the Research Community*, pages 84–89. Springer International Publishing, Cham, 2017.

[7] Kleanthi Georgala, Mirko Spasić, Milos Jovanovik, Henning Petzka, Michael Röder, and Axel-Cyrille Ngonga Ngomo. *MOCHA2017: The Mighty Storage Challenge at ESWC 2017*, pages 3–15. Springer International Publishing, Cham, 2017.

[8] Vincenzo Gulisano, Zbigniew Jerzak, Roman Katerinenko, Martin Strohbach, and Holger Ziekow. The debs 2017 grand challenge. In *Proceedings of the 11th ACM International Conference on Distributed and Event-based Systems*, DEBS '17, pages 271–273, New York, NY, USA, 2017. ACM.

[9] Dimitrije Jankov, Sourav Sikdar, Rohan Mukherjee, Kia Teymourian, and Chris Jermaine. Real-time high performance anomaly detection over data streams: Grand challenge. In *Proceedings of the 11th ACM International Conference on Distributed and Event-based Systems*, DEBS '17, pages 292–297, New York, NY, USA, 2017. ACM.

[10] Christian Mayer, Ruben Mayer, and Majd Abdo. Streamlearner: Distributed incremental machine learning on event streams: Grand challenge. In *Proceedings of the 11th ACM International Conference on Distributed and Event-based Systems*, DEBS '17, pages 298–303, New York, NY, USA, 2017. ACM.

[11] Axel-Cyrille Ngonga Ngomo, Alejandra García-Rojas, and Irini Fundulaki. HOBBIT: Holistic Benchmarking of Big Linked Data. *ERCIM News*, 2016(105), 2016.

[12] Axel-Cyrille Ngonga Ngomo, Michael Röder, Diego Moussallem, Ricardo Usbeck, and René Speck. Automatic generation of benchmarks for entity recognition and linking. *CoRR*, abs/1710.08691, 2017.

[13] Axel-Cyrille Ngonga Ngomo, Norman Heino, Klaus Lyko, René Speck, and Martin Kaltenböck. SCMS - Semantifying content management systems. In *International Semantic Web Conference*, 2011.

[14] Julien Plu, Raphaël Troncy, and Giuseppe Rizzo. *ADEL@OKE 2017: A Generic Method for Indexing Knowledge Bases for Entity Linking*, pages 49–55. Springer International Publishing, Cham, 2017.

[15] Alexander Potocki, Daniel Hladky, and Martin Voigt. *Challenge Accepted: QUAD Meets MOCHA2017*, pages 16–20. Springer International Publishing, Cham, 2017.

[16] Nikolay Radoev, Mathieu Tremblay, Michel Gagnon, and Amal Zouaq. *AMAL: Answering French Natural Language Questions Using DBpedia*, pages 90–105. Springer International Publishing, Cham, 2017.

[17] Nicolo Rivetti, Yann Busnel, and Avigdor Gal. Flinkman: Anomaly detection in manufacturing equipment with apache flink: Grand challenge. In *Proceedings of the 11th ACM International Conference on Distributed and Event-based Systems*, DEBS '17, pages 274–279, New York, NY, USA, 2017. ACM.

[18] T. Saveta, E. Daskalaki, G. Flouris, I Fundulaki, M. Herschel, and A.-C. Ngonga Ngomo. Pushing the limits of instance matching systems: A semantics-aware benchmark for linked data. In *WWW*, pages 105–106. ACM, 2015. Poster.

[19] Tzanina Saveta, Evangelia Daskalaki, Giorgos Flouris, Irini Fundulaki, and Axel-Cyrille Ngonga Ngomo. Lance: A generic benchmark generator for linked data. In *International Semantic Web Conference (Posters & Demos)*, 2015.

[20] Mohamed Ahmed Sherif, Kevin Dreßler, Panayiotis Smeros, and Axel-Cyrille Ngonga Ngomo. Radon-rapid discovery of topological relations. In *AAAI*, pages 175–181, 2017.

[21] Panayiotis Smeros and Manolis Koubarakis. Discovering spatial and temporal links among rdf data. In *LDOW@ WWW*, 2016.

[22] Daniil Sorokin and Iryna Gurevych. *End-to-End Representation Learning for Question Answering with Weak Supervision*, pages 70–83. Springer International Publishing, Cham, 2017.

[23] Mirko Spasić and Milos Jovanovik. *MOCHA 2017 as a Challenge for Virtuoso*, pages 21–32. Springer International Publishing, Cham, 2017.

[24] René Speck and Axel-Cyrille Ngonga Ngomo. *Ensemble Learning for Named Entity Recognition*, pages 519–534. Springer International Publishing, Cham, 2014.

[25] René Speck, Michael Röder, Sergio Oramas, Luis Espinosa-Anke, and Axel-Cyrille Ngonga Ngomo. *Open Knowledge Extraction Challenge 2017*, pages 35–48. Springer International Publishing, Cham, 2017.

[26] Christian Strobl. *Encyclopedia of GIS*, chapter Dimensionally Extended Nine-Intersection Model (DE-9IM), pages 240–245. Springer, 2008.

[27] Ricardo Usbeck, Axel-Cyrille Ngonga Ngomo, Bastian Haarmann, Anastasia Krithara, Michael Röder, and Giulio Napolitano. *7th Open Challenge on Question Answering over Linked Data (QALD-7)*, pages 59–69. Springer International Publishing, Cham, 2017.

[28] Ricardo Usbeck, Axel-Cyrille Ngonga Ngomo, Michael Röder, Daniel Gerber, Sandro Athaide Coelho, Sören Auer, and Andreas Both. *AGDISTIS - Graph-Based Disambiguation of Named Entities Using Linked Data*, pages 457–471. Springer International Publishing, Cham, 2014.

[29] Ricardo Usbeck, Michael Röder, Axel-Cyrille Ngonga Ngomo, Ciro Baron, Andreas Both, Martin Brümmer, Diego Ceccarelli, Marco Cornolti, Didier Cherix, Bernd Eickmann, Paolo Ferragina, Christiane Lemke, Andrea Moro, Roberto Navigli, Francesco Piccinno, Giuseppe Rizzo, Harald Sack, René Speck, Raphaël Troncy, Jörg Waitelonis, and Lars Wesemann. GERBIL – General entity annotation benchmark framework. In *24th International Conference on World Wide Web*, pages 1133–1143, 2015.

[30] Tarek Zaarour, Niki Pavlopoulou, Souleiman Hasan, Umair ul Hassan, and Edward Curry. Automatic anomaly detection over sliding windows: Grand challenge. In *Proceedings of the 11th ACM International Conference on Distributed and Event-based Systems*, DEBS '17, pages 310–314, New York, NY, USA, 2017. ACM.

[31] Lei Zou, Ruizhe Huang, Haixun Wang, Jeffrey Xu Yu, Wenqiang He, and Dongyan Zhao. Natural language question answering over RDF: A graph data driven approach. In *ACM SIGMOD International Conference on Management of Data*, pages 313–324, 2014.