A Performance Evaluation of OWL 2 DL Reasoners using ORE 2015 and Very Large Bio Ontologies

An Ngoc Lam[1,*], Brian Elvesæter[1] and Francisco Martin-Recuerda[1]

[1]*SINTEF AS, Forskningsveien 1, 0373 Oslo, Norway*

Abstract

In this paper, we evaluate the reasoning performance of six prominent OWL 2 DL reasoners, using two collections of ontologies: ORE 2015 and the 21 largest ontologies from the NCBO BioPortal. We observed that the majority of the reasoners were unable to successfully perform over half of the reasoning tasks in the NCBO BioPortal dataset which includes some very large ontologies. Despite of being a representative selection of the state-of-the-art OWL 2 DL reasoners, it came to our attention that many of them are no longer being actively maintained. These findings serve as a cautionary message to the OWL 2 reasoning community, emphasizing the need to focus research efforts towards keeping up with the growing requirements posed by new and very large ontologies and knowledge graphs.

Keywords

OWL 2, Ontology, OWL 2 Reasoners, Performance Evaluation

1. Introduction

The notion of "Knowledge Graph" was coined by Google when they introduced a new service to improve web search capabilities, where information was represented and stored as labeled graphs[1]. Since then, Knowledge Graphs have gained significant attention from both industry and academia. In short, a Knowledge Graph is a machine-readable collection of knowledge represented as a labeled graph. A Knowledge Graph contains entities, such as people and locations, and relations between them, such as "birthplace" and "work address". Relevant examples of knowledge graphs include Wikidata [1], DBpedia [2], and YAGO [3]. These knowledge graphs, among many others, are defined using the Web Ontology Language version 2 (OWL 2) [4], which is a recommendation from the W3C. The OWL 2 specification defines two semantics: (1) Direct Semantics and (2) RDF-based semantics. The former is known as OWL 2 DL and it is decidable. The OWL 2 specification also includes three tractable fragments with polynomial reasoning time (EL, QL, RL). To store, exchange, and query knowledge graphs, the W3C provides two additional recommendations: RDF 1.1 [5] and SPARQL 1.1 [6]. Triplestores are graph databases that support these two recommendations. The evaluation presented in [7]

DMKG2023: 1st International Workshop on Data Management for Knowledge Graphs, May 28, 2023, Hersonissos, Greece

[*]Corresponding author.
an.lam@sintef.no (A. N. Lam); brian.elvesater@sintef.no (B. Elvesæter); francisco.martin-recuerda@sintef.no (F. Martin-Recuerda)
0000-0003-4929-054X (A. N. Lam); 0000-0001-7304-4950 (B. Elvesæter); 0000-0002-7146-692X (F. Martin-Recuerda)

[1]https://blog.google/products/search/introducing-knowledge-graph-things-not/

demonstrated that state-of-the-art triplestores can handle very large knowledge graphs such as Wikidata. However, most of these triplestores do not support OWL 2 DL, or they only support one of the tractable fragments, such as OWL 2 RL. Apache Fuseki[2] is an example of the former case, and RDFox[3] is an example of the latter case.

Because many triplestores do not support OWL 2 DL or only one of its fragments, it may not be possible to guarantee the correctness of queries if the knowledge graph includes certain OWL 2 constructors. This could be the case with the knowledge graph YAGO 4 [3], which defines the 6 top classes as disjoint classes, meaning they cannot share any individuals. A careless extension of YAGO 4 may easily violate these design restrictions, and only an OWL 2 DL reasoner might be able to identify these violations. Therefore, it might be recommended to adopt an OWL 2 DL reasoner when working with OWL 2 DL knowledge graphs. However, as many of the available reasoners were designed to test different reasoning algorithms, optimizations, and extensions beyond OWL 2 DL, deciding on an appropriate reasoner for a particular application can be challenging.

In this paper, we evaluate the performance of six prominent OWL 2 DL reasoners, namely Pellet [8], FaCT++ [9], JFact[4], Openllet[5], HermiT [10], and Konclude [11]. Except for Konclude, these reasoners support the OWLAPI library[6], which is one of the most relevant ontology and semantic knowledge graph development frameworks. To test these reasoners, we selected two collections of OWL 2 ontologies: ORE 2015 [12] and the 21 largest ontologies from the NCBO BioPortal [13]. The evaluation scripts and results of this work can be found at the GitHub repository[7].

The remainder of the paper is structured as follows. Section 2 discusses related work. Section 3 describes the evaluation setup. Section 4 provides a detailed discussion of the evaluation results. Finally, Section 5 concludes the paper and presents future work directions.

2. Related Work

During the past two decades, a number of benchmarks and frameworks have been introduced to support the comparison and evaluation of the reasoners. Some of these benchmarks were based on generating synthetic datasets such as [14, 15, 16, 17, 18]. These early initiatives were used to test several representative reasoners at that time on a relatively low number of ontologies and a small set of hand-crafted queries [19]. The LUBM benchmark [20], together with its extensions - UOBM [21] and SLUBM [22], are the most representative and popular synthetic benchmarks with varying sizes of instances for the university ontology. However, due to several shortcomings in their design (e.g., sparsely interrelated data and very simple schemas), these cannot be considered as meaningful benchmarks as many reasoners that achieved excellent results in these benchmarks failed other ABox or TBox tests [23]. In contrast, using real-world ontologies to evaluate the performance of the reasoners was a different direction where the

[2]https://jena.apache.org/documentation/fuseki2/
[3]https://www.oxfordsemantic.tech/product
[4]https://github.com/owlcs/jfact
[5]https://github.com/Galigator/openllet
[6]https://github.com/owlcs/owlapi
[7]https://github.com/SINTEF-9012/owl-reasoner-evaluation

results can be easily interpreted and exploited [24]. In this light, several benchmarks and evaluations were introduced in [24, 25, 26, 27]. However, these works used just a small set of ontologies and thus cannot be used to test the scalability of the reasoner. The ORE benchmark which was a part of OWL Reasoner Evaluation (ORE) Competition [12] contains a large dataset of ontologies of varying sizes sampled from different ontology libraries. This benchmark can be used to evaluate the reasoners in terms of the coverage of OWL 2 constructors and scalability.

This work is motivated by the ORE competition in order to provide a thorough understanding of state-of-the-art OWL 2 DL reasoners to assist developers in the selection of appropriate solutions for their semantic applications. We focus only on reasoning performances as there were several comprehensive surveys and comparisons of reasoners with regard to other aspects such as query language, programming interface and OWL 2 constructors coverage as in [18, 28, 29]. In this light, a new evaluation of state-of-the-art reasoners with the stress on scalability and large-scale knowledge graph is in need. There are several new releases of the reasoners since the last ORE 2015 competition, while the most recent benchmark papers, such as OWL2Bench (2020) [30] and the evaluation (2022) presented in [19], did not include all well-known reasoners or focused mainly on energy profiling. Due to these shortcomings, we decided to conduct a new performance evaluation using the most recent versions of the reasoners with the ORE 2015 dataset. In addition, in order to investigate how these reasoners can perform on very large and modern ontologies, we selected the 21 largest ones from the NCBO BioPortal.

3. Evaluation Setup

The evaluation was conducted on Amazon Elastic Compute Cloud (EC2) `r5.2xlarge`[8] instances equipped with 8 vCPUs of Intel Xeon Platinum 8000 series processor at up to 3.1 GHz, 64 GB RAM, Ubuntu 18.04 LTS, and 100 GB EBS `gp3` volume which offers SSD-performance with predictable, baseline performance of 3,000 IOPS and 125 MB/s throughput[9].

3.1. Reasoners

We aim to evaluate the performance of state-of-the-art reasoners in order to support the selection of appropriate libraries for semantic data processing and reasoning applications. In this regard, we selected only OWL 2 DL reasoners that are still actively maintained within the last 10 years. The reasoners should provide interfaces for OWLAPI (either version 4 or 5) - the ontology development framework supported by most reasoners [29] - in order to enable the integration into our benchmark programs. Based on those criteria, the following six reasoners: Pellet, FaCT++, JFact, Openllet, HermiT, and Konclude are selected.

Table 1 lists the latest releases (before March 2023) of the evaluated reasoners. Except for Pellet and FaCT++ which support OWLAPI 4, the others provide interfaces for the latest OWLAPI 5. Furthermore, as Konclude only provides a standalone version with support for OWLLink protocol[10], the OWLLink-OWLAPI library was used to integrate the reasoner into OWLAPI.

[8]https://aws.amazon.com/ec2/instance-types/r5/
[9]https://aws.amazon.com/ebs/general-purpose/
[10]https://www.w3.org/Submission/owllink-structural-specification/

Table 1
List of the evaluated reasoners.

Reasoners	Version	OWLAPI Version	Release Date	Repository
Pellet	2.4.0	4.5.20	18.4.2015	https://github.com/ignazio1977/pellet
FaCT++	1.6.5	4.5.20	31.12.2016	https://bitbucket.org/dtsarkov/factplusplus
JFact	5.0.3	5.1.19	23.09.2018	https://github.com/owlcs/jfact
Openllet	2.6.5	5.1.19	27.09.2019	https://github.com/Galigator/openllet
HermiT	1.4.5.519	5.1.19	18.02.2020	https://github.com/owlcs/hermit-reasoner
OWLLink	2.0.0	5.1.19	21.12.2018	https://github.com/ignazio1977/owllink-owlapi
Konclude	0.7.0-1138	5.1.19	19.06.2021	https://github.com/konclude/Konclude

3.2. Datasets

The evaluation has been carried out on two datasets: ORE 2015 [12] and NCBO BioPortal [13] ontologies. The ontologies are binned by size into the following groups: very small (less than 1K axioms), small (1K - 10K axioms), medium (10K - 100K axioms), large (100K - 1M axioms), very large (1M - 10M axioms), and huge (more than 10M axioms), as illustrated in Figure 1.

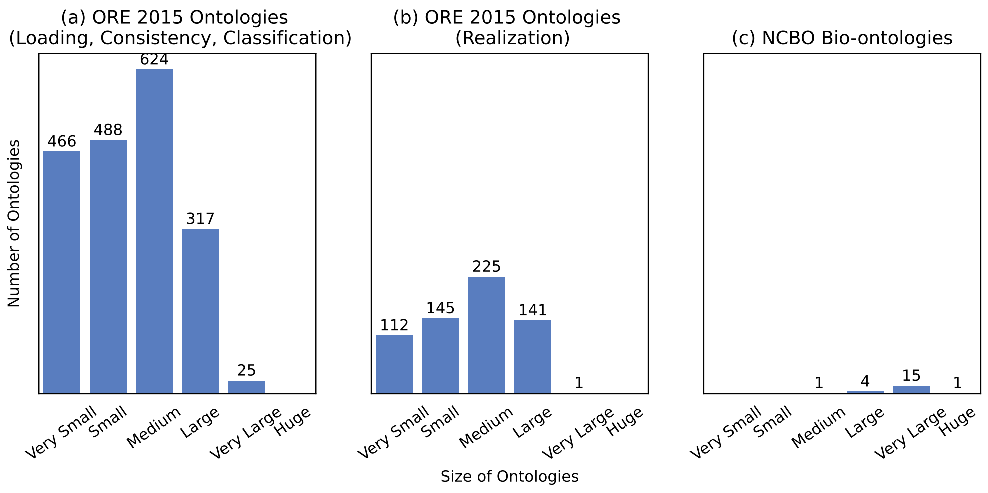

Figure 1: Distribution of the ontologies in each dataset.

ORE 2015 contains 1920 ontologies from the OWL Reasoner Evaluation 2015 Competition dataset[11], which can be used to evaluate the performance of the reasoners on small and medium-size ontologies. Regarding the reasoning task realization, we only consider ontologies with more than 100 ABox axioms. This process results in a subset of 624 ontologies for the realization task.

NCBO BioPortal is one of the largest libraries of biomedical ontologies and terminologies. Ontologies have been selected by taking the 20 largest ones in this portal. Additionally, the GALEN ontology [31] was also added to this dataset as it is known to be one of the most difficult ontologies for tableau reasoners. This ontology contains many cyclic axioms, which result

[11]https://zenodo.org/record/18578

in very large models and an extremely long time to classify and realize [32]. Table 2 shows the statistics of those 21 ontologies. This dataset is used to evaluate the performance of the reasoners on very large real-world ontologies and knowledge graphs.

Table 2
NCBO Bio-ontologies statistics: number of Classes, Ind[ividuals], Axioms, Logical Axioms, TBox, ABox, Data Prop[erties], and Obj[ect] Prop[erties]. The list were ordered by total number of axioms.

Ontology	Classes	Ind	Axioms	Logical Axioms	TBox	ABox	Data Prop	Obj Prop
GALEN	23,136	0	63,324	37,696	36,205	0	0	950
HOOM	117,829	1	461,823	222,011	221,980	1	2	12
OCHV	115,645	0	505,642	173,501	173,475	0	1	17
FMA	104,721	2	791,248	296,460	296,329	2	29	139
SNMI	109,156	109,150	998,278	490,449	109,047	381,376	6	21
UPHENO	159,212	17,091	1,552,159	468,679	446,500	21,697	1	341
RHMESH	305,349	0	1,654,092	432,805	432,805	0	0	1
RETO	147,738	138,523	1,927,495	941,797	152,173	789,624	0	12
IOBC	126,847	66,923	1,929,343	316,724	182,277	134,384	0	59
NIFSTD	157,666	130,781	1,962,343	483,775	320,798	161,717	6	833
REXO	158,239	149,084	2,043,111	997,563	162,839	834,724	0	12
GEXO	166,254	157,102	2,143,045	1,052,276	170,932	881,344	0	12
NCIT	169,111	0	2,850,463	234,117	234,104	0	0	97
CHEBI	171,059	50,642	3,314,948	397,690	310,133	87,554	0	10
DRON	578,206	510,104	3,498,249	1,642,634	1,158,027	484,583	1	20
PR	332,354	294,425	5,151,763	1,426,166	1,007,050	419,110	0	21
LOINC	268,558	268,552	7,079,178	5,632,197	257,534	5,374,637	70	105
GAZ	668,839	660,755	7,801,093	2,256,702	1,460,125	796,571	0	15
MESH	347,698	347,692	9,612,820	7,289,654	40,919	7,248,709	28	27
CCO	264,892	255,182	9,925,469	4,229,990	2,249,881	1,980,102	0	15
NCBITAXON	1,908,229	1,908,223	17,351,973	9,541,037	1,908,189	7,632,822	3	18

3.3. Experiment Configuration

To run the experiment, we implemented two small Java programs using OWLAPI to: (1) perform a reasoning task *T* on an ontology *O* using a particular reasoner *R*, and (2) run the evaluation with the sets of reasoners *Rs*, ontologies *Os*, and tasks *Ts*. By splitting the evaluation into two different programs, each reasoning task for a reasoner on an ontology can be run in a separate process. Particularly, for each evaluation iteration, the ontology is loaded again, and a new instance of the reasoner for the corresponding ontology is created. Therefore, this setup eliminates the effect of memory limitation and caching on the measured execution times. The default settings were applied for all reasoners as we assume that the general users do not have too much knowledge to optimize the configurations. There are four different tasks were evaluated:

- **Loading reasoner**: regards the time for initializing a new instance of the reasoner and loading the ontology model into it.
- **Consistency**: verifies whether every class in the ontology admits at least one individual.
- **Classification**: computes a class hierarchy with all superclasses and subclasses of every class defined in an ontology.
- **Realization**: for each individual, it finds all classes, especially the most specific ones, where the individual is an instance of.

The timeout was set to 30 minutes for ORE 2015 dataset and 1 hour for BioPortal ontologies. Our primary performance measure is the task execution time. Each reasoning task was evaluated for 10 iterations, and the mean values were used for the analysis. We propose to use both arithmetic mean M_a and geometric mean M_g (n^{th} root of the product over n numbers) of the execution time. To compute the means, the failed tasks (e.g., timeout, errors) are penalized with double the timeout value (1 hour for ORE 2015 and 2 hours for BioPortal ontologies). Arithmetic mean is dominated by large numbers, so it can be used as an indicator of a high ratio of success and failure tasks (i.e., smaller value indicates more success tasks). Geometric mean can handle varying proportions and mitigate the effect of large number penalty. Therefore, geometric mean is used to evaluate the overall performance over success tasks (i.e., smaller value indicates smaller execution time for the success tasks).

4. Results and Discussions

4.1. ORE 2015 Dataset

Table 3 presents the results of the evaluated reasoners on ORE 2015 dataset. Additionally, the task reasoning times are plotted in Figure 2, where timeouts and errors were excluded.

Table 3
Results of the reasoning tasks (in seconds) on ORE 2015 ontologies. (TO: Timeout, OOM: OutOfMemory Error, OE: Other Errors, IE: Inconsistent Error, #IO: Number of Inconsistent Ontologies).

	Rank	**Reasoners**	**Success**	**TO**	**OOM**	**OE**	M_a	M_g	
Loading	1	JFact	1920	0	0	0	**0.35**	**0.06**	
	2	Openllet	1918	2	0	0	6.05	0.12	
	3	Konclude	1916	3	0	1	10.04	0.28	
	4	HermiT	1890	0	1	29	57.29	0.21	
	5	FaCT++	1878	0	0	42	79.11	**0.06**	
	6	Pellet	1641	2	0	277	525.5	0.51	
	Rank	**Reasoners**	**Success**	**TO**	**OOM**	**OE**	**#IO**	M_a	M_g
Consistency	1	Konclude	1911	4	0	5	43	**16.88**	**0.00**
	2	HermiT	1881	7	3	29	43	79.68	0.01
	3	Openllet	1818	93	5	4	58	195.32	0.09
	4	FaCT++	1800	31	0	89	24	234.74	0.05
	5	JFact	1781	110	0	29	69	283.32	0.59
	6	Pellet	1513	84	3	320	33	766.28	0.52
	Rank	**Reasoners**	**Success**	**TO**	**OOM**	**IE**	**OE**	M_a	M_g
Classification	1	Konclude	1862	7	0	43	8	**109.65**	**0.05**
	2	HermiT	1751	84	12	43	30	346.70	2.21
	3	FaCT++	1598	204	0	24	94	628.41	0.51
	4	Openllet	1595	216	12	59	38	630.27	1.89
	5	JFact	1486	327	1	70	36	862.23	6.79
	6	Pellet	1315	198	10	33	364	1158.64	5.67
	Rank	**Reasoners**	**Success**	**TO**	**OOM**	**IE**	**OE**	M_a	M_g
Realization	1	Konclude	591	3	0	25	5	**190.87**	**0.15**
	2	HermiT	512	68	2	25	17	708.55	15.44
	3	Openllet	488	84	8	36	8	806.45	5.24
	4	FaCT++	471	79	0	14	60	907.23	1.62
	5	JFact	439	140	1	36	8	1124.64	18.65
	6	Pellet	339	81	5	22	177	1661.65	31.03

Regarding loading task, JFact ranked first with no error or timeout, while Openllet and

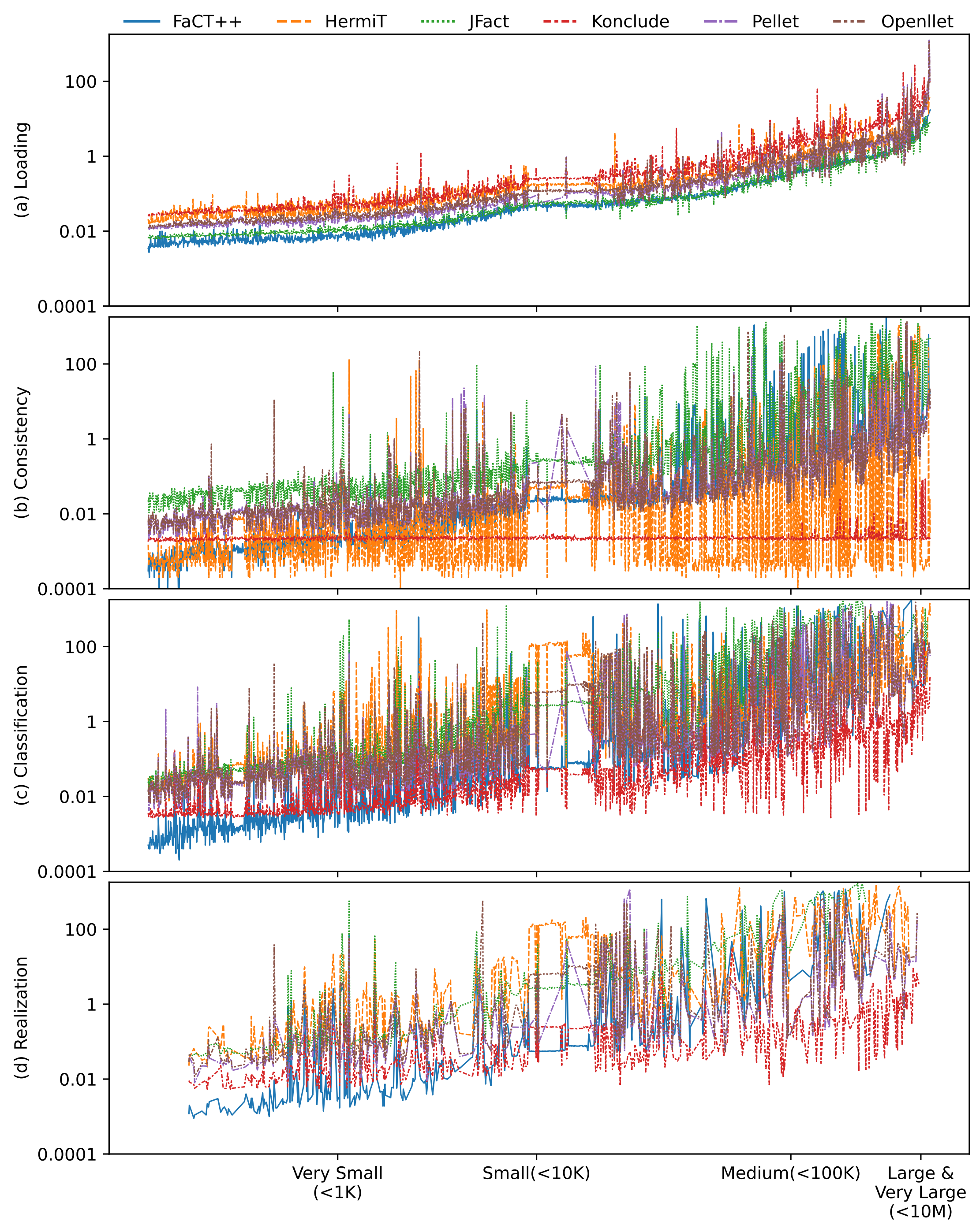

Figure 2: Reasoning time in seconds (y-axis) of ORE 2015 ontologies (x-axis) ordered by the number of axioms.

Konclude are in the top three with up to 3 timeouts. Although Openllet was developed based on Pellet, the reasoner had much better results with no errors, while Pellet had the most errors. Konclude had one error related to OWLLINK library (i.e., not supported `DatatypeDefinition`). HermiT, FaCT++, and Pellet did not succeed in loading all ontologies due to a lot of errors. Most of the errors are related to OWL constructs (e.g., SWRL rule uses a built-in atom, unsupported datatype). Regarding the execution time, JFact and FaCT++ seem to load ontologies faster than the others as they have the smallest geometric mean, while Konclude amounted to the largest value of geometric mean. As OWLLink library also validates consistency after loading ontologies, Konclude took a longer time to complete this initialization and loading task while having much better results in the consistency validation task. This insight can also be noticed in Figure 2(a), where Konclude usually lies above others while FaCT++ and JFact are the lowest.

Regarding the consistency validation task, Konclude and HermiT were in the top two, with the least timeouts and errors. Although JFact was loaded fastest, it had the most timeouts in this task. JFact also had the largest geometric mean, indicating that the reasoners had the poorest performance on this task. Furthermore, Pellet ranks last due to many errors related to unsupported datatype (17% of the ontologies), while its descendant, Openllet, is in the fourth position with only 4 errors. Regarding the overall performance of the reasoners on success ontologies, Konclude has the best performance with the smallest geometric mean. As seen from Figure 2(b), Konclude has approximately constant performance on this task because consistency checking was already done when loading the reasoner, as discussed earlier. On the other hand, HermiT can perform even better than Konclude on many small ontologies. Furthermore, it is worth mentioning that although Openllet and Pellet vary a lot in the number of errors, these two reasoners have very similar performance on all reasoning tasks on success ontologies. This can be seen in Figure 2(b), where Openllet lies just slightly above Pellet in all four plots. This finding shows that the Openllet reasoner was developed to resolve the limitation of Pellet on OWL features support without significant improvement in reasoning performance.

Konclude and HermiT were also in the top two reasoners for classification and realization tasks, while JFact and Pellet always ranked last when comparing the number of success ontologies. Furthermore, although FaCT++ had more failures than Konclude and HermiT, it performs better on small ontologies. This can be seen from the geometric mean as well as in 2(c) and (d), where FaCT++ had the best performance on small ontologies. Openllet and Pellet are still in the middle range, while HermiT performed slower than these two reasoners, although HermiT had much fewer timeouts. JFact still had the poorest performance.

In summary, Konclude and HermiT were always in the top two reasoners with the most successful reasoning tasks on the ORE 2015 ontologies, although they did not always have the best reasoning times. Specifically, Konclude is more scalable as the reasoner had the best performance on large and very large ontologies. HermiT had very good performance on consistency validation. However, for classification and realization tasks, this reasoner was outperformed by the other reasoners. Therefore, it is most suitable for applications that require only consistency validation. Furthermore, FaCT++ was the next reasoner in the list with more timeouts and errors than HermiT but had much better performance on small ontologies. This reasoner can be considered for applications with small knowledge graphs or ontologies with simple OWL constructs. Pellet and its descendant Openllet were always in the middle range with slight slower performance than FaCT++. Furthermore, Openllet was developed based on

Pellet with substantial improvements related to OWL syntax support. However, regarding task execution performance, Openllet and Pellet had very similar results. Finally, JFact consistently ranked last in the evaluation.

4.2. NCBO BioPortal Dataset

Table 4
Reasoning time (in seconds) for NCBO Bio-ontologies. Ontologies were ordered by number of axioms.

Ontology	Loading						Consistency					
	JFact	Openllet	Pellet	FaCT++	HermiT	Kc	JFact	Openllet	Pellet	FaCT++	HermiT	Kc
GALEN	**0.25**	1.16	0.93	0.28	2.73	1.88	OE	0.54	0.48	37.27	**0.00**	**0.00**
HOOM	**1.88**	5.68	4.88	12.15	22.35	23.43	TO	TO	TO	TO	0.02	**0.00**
OCHV	1.23	2.66	1.98	**1.16**	3.26	5.25	209.05	0.44	0.45	0.61	**0.00**	**0.00**
FMA	**1.46**	1721.50	1726.70	1.81	11.72	11.72	OE	3.92	3.90	OE	**0.00**	**0.00**
SNMI	**3.08**	7.87	6.83	3.39	5.07	15.51	TO	1.40	1.32	3.18	7.13	**0.00**
UPHENO	**3.18**	11.44	13.81	3.59	OOM	119.07	TO	TO	TO	TO	OOM	**0.00**
RHMESH	3.64	7.58	6.45	**3.35**	8.21	10.93	168.32	1.27	1.22	1.70	**0.00**	**0.00**
RETO	**4.35**	10.95	9.28	5.20	7.25	27.58	738.17	1.39	1.44	3.90	2.37	**0.00**
IOBC	2.97	4.83	4.77	**2.36**	4.29	9.35	318.15	1.22	1.14	1.88	1.66	**0.00**
NIFSTD	**4.41**	11.75	OE	OE	OE	25.99	TO	TO	OE	OE	OE	**0.05**
REXO	**5.16**	11.66	10.15	5.31	7.81	29.09	839.53	1.40	1.41	4.20	2.39	**0.00**
GEXO	**5.33**	12.32	10.64	5.87	8.25	31.04	905.30	1.34	1.38	4.59	2.57	**0.00**
NCIT	**2.96**	5.94	5.99	OE	16.48	OE	255.83	0.77	0.77	OE	**0.00**	OE
CHEBI	3.44	8.00	8.04	**2.70**	6.14	12.80	675.13	1.15	1.16	2.36	0.18	**0.01**
DRON	17.26	37.31	27.12	**14.82**	32.36	63.35	TO	14.47	13.45	11.14	13.43	**0.00**
PR	11.36	22.38	19.69	**9.99**	28.46	55.89	TO	TO	1208.59	OE	1.46	**0.00**
LOINC	**16.46**	446.04	OE	OE	34.14	OOM	TO	**5.55**	OE	OE	40.25	OOM
GAZ	22.51	48.49	37.45	**16.95**	33.14	71.27	TO	8.71	8.91	12.42	3.51	**0.00**
MESH	**19.57**	653.98	OE	OE	47.73	OOM	TO	**9.47**	OE	OE	73.48	OOM
CCO	**14.53**	107.53	103.42	21.91	39.94	OOM	TO	12.49	13.05	23.09	**8.14**	OOM
NCBITAXON	**76.58**	211.58	186.09	89.40	121.53	OOM	TO	30.57	**27.67**	71.11	OOM	OOM

Ontology	Classification						Realization					
	JFact	Openllet	Pellet	FaCT++	HermiT	Kc	JFact	Openllet	Pellet	FaCT++	HermiT	Kc
GALEN	OE	TO	TO	TO	OOM	**8.90**	OE	TO	TO	TO	**0.00**	9.37
HOOM	TO	TO	TO	TO	TO	184.37	TO	TO	TO	TO	TO	164.23
OCHV	TO	744.70	869.32	1699.32	28.02	**0.78**	TO	963.34	832.12	1822.65	**0.00**	1.52
FMA	OE	OE	OE	OE	TO	**7.37**	OE	OE	OE	OE	TO	**7.68**
SNMI	TO	10.86	9.35	2105.98	225.40	**0.51**	TO	10.98	9.54	1873.32	1465.84	**1.11**
UPHENO	OE	TO	TO	TO	OOM	OE	TO	TO	TO	TO	OOM	OE
RHMESH	TO	8.59	8.16	7.76	23.86	**2.88**	TO	8.71	8.27	7.13	**0.00**	4.72
RETO	TO	3.24	3.27	2296.49	11.55	**0.70**	TO	3.28	3.24	2318.47	18.54	**1.52**
IOBC	IE	165.83	170.01	TO	256.41	**1.85**	IE	173.81	172.97	TO	1216.44	**2.53**
NIFSTD	TO	TO	OE	OE	OE	IE	TO	TO	OE	OE	OE	IE
REXO	TO	3.43	3.34	2768.20	12.14	**0.75**	TO	3.48	3.35	2537.54	20.19	**1.62**
GEXO	TO	3.50	3.46	3283.48	12.75	**0.78**	TO	3.55	3.59	2890.13	19.83	**1.69**
NCIT	TO	202.69	733.82	OE	**157.33**	OE	TO	215.85	OE	OE	**0.00**	OE
CHEBI	TO	1042.31	1038.34	2439.78	135.77	**2.10**	TO	837.44	1152.21	2286.37	152.40	**2.96**
DRON	TO	TO	TO	TO	TO	35.88	TO	TO	TO	TO	TO	39.13
PR	TO	TO	TO	OE	TO	32.66	TO	TO	TO	OE	TO	24.11
LOINC	TO	TO	OE	OE	1908.58	OOM	TO	TO	OE	OE	TO	OOM
GAZ	TO	TO	TO	TO	1260.37	**9.95**	TO	TO	TO	TO	TO	**13.81**
MESH	TO	TO	OE	OE	TO	OOM	TO	TO	OE	OE	TO	OOM
CCO	TO	TO	TO	TO	OOM	OOM	TO	TO	TO	TO	OOM	OOM
NCBITAXON	TO	357.16	271.79	TO	OOM	OOM	TO	343.02	282.06	TO	OOM	OOM

The results of the reasoners' evaluation on NCBO BioPortal are presented in Table 4. To evaluate the performance of the reasoners on large-scale knowledge graphs, we selected the 21 largest ontologies. The results for the reasoner loading task are consistent with those of ORE 2015, where JFact was the fastest reasoner and Konclude (Kc) ranked last due to additional time required for consistency validation. Konclude also had the highest number of failures, which were attributed to memory overflow. Upon closer examination of these OOM errors, we found that they were related to the OWLLINK library, which is not designed to handle very large ontologies. However, when we used KoncludeCLI - the standalone version of Konclude - to perform the reasoning tasks on the same ontologies, we did not encounter any errors. Additionally, the table shows that Openllet performed slightly better than its previous version - Pellet - and was also capable of handling several syntax errors that occurred with Pellet.

Konclude performed exceptionally well in the consistency task, owing to its ability to validate consistency during the reasoner loading process. However, when we compared the loading time of Konclude with the consistency validation time of the other reasoners, HermiT emerged as the top-performing reasoner in this test. This outcome aligns with the results of the ORE 2015 evaluation, which also found HermiT to be superior in checking ontology consistency, particularly with very large ontologies.

In terms of classification and realization, Konclude outperformed the other reasoners overall. JFact had the poorest performance, failing on all ontologies in this evaluation. Additionally, HermiT had fewer errors than FaCT++, Openllet, and Pellet. However, when considering only the successful cases, HermiT appeared to run slower than Openllet and Pellet, a finding consistent with the results of ORE 2015. Finally, FaCT++ exhibited inferior performance on these two reasoning tasks, suggesting it may not be efficient for very large ontologies.

Overall, the results of the NCBO bio-ontologies evaluation were consistent with the findings of the ORE 2015 evaluation. Konclude emerged as the top-performing reasoner, while JFact failed on nearly all ontologies. Notably, Konclude was the only reasoner to successfully complete all tasks on the GALEN ontology, which is considered one of the most difficult ontologies due to its many cyclic axioms. However, Konclude may encounter memory overflow errors when performing reasoning tasks on some very large ontologies, due to limitations with OWLLINK library. This issue does not arise when using KoncludeCLI. HermiT was the second-best performing reasoner with fewer errors and decent execution results, followed by Openllet, Pellet, and FaCT++, all of which had failures on almost half of the ontologies.

5. Conclusion and Future Work

This paper presents a comprehensive evaluation of the state-of-the-art OWL 2 DL reasoners using two real-world ontology datasets from ORE 2015 and NCBO BioPortal. The evaluation results from both datasets were consistent, with Konclude and HermiT frequently ranking at the top for successful reasoning tasks, while JFact had the most failures. In terms of reasoning time, Konclude demonstrated superior performance, while the others showed varying performance across different reasoning tasks and ontology sizes. Our evaluation identifies the strengths and weaknesses of each reasoner, providing valuable insights for ontology developers and application designers when selecting the most suitable reasoning system for their specific needs.

During the evaluation of the reasoners, it came to our attention that many of them are no longer being actively maintained. This is apparent from Table 1, which shows that several reasoners have not been updated in the past five years. Additionally, the majority of reasoners were unable to successfully perform over half of the reasoning tasks in the NCBO BioPortal dataset, which consists of large to very large ontologies. These findings serve as a cautionary message to the OWL 2 reasoning community, emphasizing the need to focus research efforts towards keeping up with the ever-evolving complexity of the ontology language and the growing requirements of semantic applications.

As for future work, we aim to expand this evaluation by incorporating large and prominent OWL 2 knowledge graphs such as DBpedia, Wikidata or YAGO. These knowledge graphs have gained significant attention across various domains and have been adopted in many applications.

Acknowledgments

The authors would like to thank the anonymous reviewers for their valuable feedback. This work has been funded by The Research Council of Norway projects SkyTrack (No 309714), DataBench Norway (No 310134) and SIRIUS Centre (No 237898), and the European Commission projects DataBench (No 780966), VesselAI (No 957237), Iliad (No 101037643), enRichMyData (No 101070284), Circular TwAIn (No 101058585), and Graph-Massivizer (No 101093202).

References

[1] D. Vrandečić, M. Krötzsch, Wikidata: A Free Collaborative Knowledgebase, Communications of the ACM 57 (2014) 78–85.

[2] J. Lehmann, R. Isele, M. Jakob, A. Jentzsch, D. Kontokostas, P. N. Mendes, S. Hellmann, M. Morsey, P. Van Kleef, S. Auer, et al., DBpedia – A Large-scale, Multilingual Knowledge Base Extracted from Wikipedia, Semantic web 6 (2015) 167–195.

[3] T. Pellissier Tanon, G. Weikum, F. Suchanek, YAGO 4: A Reason-able Knowledge Base, in: The Semantic Web: 17th International Conference, ESWC 2020, Heraklion, Crete, Greece, May 31–June 4, 2020, Proceedings 17, Springer, 2020, pp. 583–596.

[4] B. Motik, P. F. Patel-Schneider, B. Parsia, C. Bock, A. Fokoue, P. Haase, R. Hoekstra, I. Horrocks, A. Ruttenberg, U. Sattler, et al., OWL 2 Web Ontology Language Structural Specification and Functional-Style Syntax, W3C recommendation (2009).

[5] D. W. Richard Cyganiak, M. Lanthaler, RDF 1.1 Concepts and Abstract Syntax, W3C recommendation (2014).

[6] S. Harris, A. Seaborne, SPARQL 1.1 Query Language, W3C recommendation (2014).

[7] A. N. Lam, B. Elvesæter, F. Martin-Recuerda, Evaluation of a Representative Selection of SPARQL Query Engines using Wikidata, in: The Semantic Web - 20th International Conference, ESWC 2023, Hersonissos, Crete, Greece, May 28 - June 1, 2023, Proceedings, Springer, 2023.

[8] E. Sirin, B. Parsia, B. C. Grau, A. Kalyanpur, Y. Katz, Pellet: A practical OWL-DL reasoner, Journal of Web Semantics 5 (2007) 51–53.

[9] D. Tsarkov, I. Horrocks, FaCT++ Description Logic Reasoner: System Description, in: International Joint Conference on Automated Reasoning, Springer, 2006, pp. 292–297.

[10] B. Glimm, I. Horrocks, B. Motik, G. Stoilos, Z. Wang, HermiT: An OWL 2 Reasoner, Journal of Automated Reasoning 53 (2014) 245–269.

[11] A. Steigmiller, T. Liebig, B. Glimm, Konclude: System description, Journal of Web Semantics 27 (2014) 78–85.

[12] B. Parsia, N. Matentzoglu, R. S. Gonçalves, B. Glimm, A. Steigmiller, The OWL Reasoner Evaluation (ORE) 2015 Competition Report, Journal of Automated Reasoning 59 (2017) 455–482.

[13] N. F. Noy, N. H. Shah, P. L. Whetzel, B. Dai, M. Dorf, N. Griffith, C. Jonquet, D. L. Rubin, M.-A. Storey, C. G. Chute, et al., Bioportal: ontologies and integrated data resources at the click of a mouse, Nucleic Acids Research 37 (2009) W170–W173.

[14] P. F. Patel-Schneider, R. Sebastiani, A New General Method to Generate Random Modal Formulae for Testing Decision Procedures, Journal of Artificial Intelligence Research 18 (2003) 351–389.

[15] J. Hladik, A Generator for Description Logic Formulas, in: Description Logics, 2005.

[16] F. Massacci, F. M. Donini, Design and Results of TANCS-2000 Non-Classical (Modal) Systems Comparison, in: Automated Reasoning with Analytic Tableaux and Related Methods: International Conference, TABLEAUX 2000, St Andrews, Scotland, UK, July 3-7, 2000 Proceedings 9, Springer, 2000, pp. 52–56.

[17] I. Horrocks, P. F. Patel-Schneider, DL Systems Comparison, in: Proc. of the 1998 Description Logic Workshop (DL'98), volume 11, 1998, pp. 55–57.

[18] J. Bock, P. Haase, Q. Ji, R. Volz, Benchmarking OWL Reasoners, in: ARea2008-Workshop on Advancing Reasoning on the Web: Scalability and Commonsense, Tenerife Spain, 2008.

[19] F. Scioscia, I. Bilenchi, M. Ruta, F. Gramegna, D. Loconte, A multiplatform energy-aware OWL reasoner benchmarking framework, Journal of Web Semantics 72 (2022) 100694.

[20] Y. Guo, Z. Pan, J. Heflin, LUBM: A benchmark for OWL knowledge base systems, Journal of Web Semantics 3 (2005) 158–182.

[21] L. Ma, Y. Yang, Z. Qiu, G. Xie, Y. Pan, S. Liu, Towards A Complete OWL Ontology Benchmark, in: The Semantic Web: Research and Applications: 3rd European Semantic Web Conference, ESWC 2006 Budva, Montenegro, June 11-14, 2006 Proceedings 3, Springer, 2006, pp. 125–139.

[22] T. N. Nguyen, W. Siberski, SLUBM: An Extended LUBM Benchmark for Stream Reasoning., in: OrdRing@ ISWC, 2013, pp. 43–54.

[23] T. Weithöner, T. Liebig, M. Luther, S. Böhm, F. Von Henke, O. Noppens, Real-World Reasoning with OWL, in: The Semantic Web: Research and Applications: 4th European Semantic Web Conference, ESWC 2007, Innsbruck, Austria, June 3-7, 2007. Proceedings 4, Springer, 2007, pp. 296–310.

[24] Z. Pan, Benchmarking DL Reasoners Using Realistic Ontologies., in: OWLED, volume 188, 2005.

[25] C. Tempich, R. Volz, Towards a benchmark for Semantic Web reasoners - an analysis of the DAML ontology library, in: EON, volume 87, 2003.

[26] T. Gardiner, D. Tsarkov, I. Horrocks, Framework For an Automated Comparison of Description Logic Reasoners, in: The Semantic Web-ISWC 2006: 5th International Semantic Web Conference, ISWC 2006, Athens, GA, USA, November 5-9, 2006. Proceedings 5, Springer, 2006, pp. 654–667.

[27] K. Dentler, R. Cornet, A. Ten Teije, N. De Keizer, Comparison of Reasoners for large Ontologies in the OWL 2 EL Profile, Semantic Web 2 (2011) 71–87.

[28] S. Abburu, A survey on ontology reasoners and comparison, International Journal of Computer Applications 57 (2012).

[29] N. Matentzoglu, J. Leo, V. Hudhra, U. Sattler, B. Parsia, A Survey of Current, Stand-alone OWL Reasoners., in: ORE, Citeseer, 2015, pp. 68–79.

[30] G. Singh, S. Bhatia, R. Mutharaju, OWL2Bench: A Benchmark for OWL 2 Reasoners, in: International Semantic Web Conference, Springer, 2020, pp. 81–96.

[31] A. Rector, J. Rogers, Ontological and Practical Issues in Using a Description Logic to Represent Medical Concept Systems: Experience from GALEN, in: Reasoning Web International Summer School, Springer, 2006, pp. 197–231.

[32] Y. Kazakov, M. Krötzsch, F. Simančík, The Incredible ELK, Journal of Automated Reasoning 53 (2014) 1–61.