LarKC

The Large Knowledge Collider
a platform for large scale integrated reasoning and Web-search

FP7 – 215535

D1.1.2

Empirical Analysis of the Initial Knowledge Representation Formalism

Florian Fischer, Barry Bishop

Document Identifier:	LarKC/2008/D1.1.2
Class Deliverable:	LarKC EU-IST-2008-215535
Version:	version 1.0
Date:	March 19, 2009
State:	final
Distribution:	public

Executive Summary

The minimal representation language for LarKC (see D1.3.1), called $L2$, deliberately has limited expressivity in order to meet its main objectives of scalability and tractability. The purpose of this deliverable is to perform an empirical analysis of these scalability properties on practical RDF repositories that not only store RDF triples, but also perform inference over them. By this means we attempt to gain insights into the feasibility of a practical implemention of $L2$ in addition to purely theoretical results. This should also help to analyse common performance pitfalls that exist even in very lightweight KR formalisms.

In this deliverable we more closely investigate the sources of computational complexity for $L2$, first by briefly observing theoretical results and furthermore by a practical evaluation. For this purpose we identify the most important performance factors for practical inference on implementations that have to be considered for a useful performance analysis of $L2$. The deliverable documents the experimental setup parameters used for the evaluation of $L2$, such as the basic hardware and software setup and the chosen data-sets. Also included is a rational for the specific choices taken. Furthermore we give a brief overview of the concrete system as well as the configuration for the experiment.

Finally, we provide the results of the analysis and the conclusions we can draw based on them, which should be helpful for the purpose of further refining $L2$ as well as for practical implementations.

Document Information

IST Project Number	FP7 – 215535	**Acronym**	LarKC
Full Title	Large Knowledge Collider		
Project URL	http://www.larkc.eu/		
Document URL			
EU Project Officer	Stefano Bertolo		

Deliverable	**Number**	1.1.2	**Title**	Empirical Analysis of the Initial Knowledge Representation Formalism
Work Package	**Number**	1	**Title**	Conceptual Framework & Evaluation

Date of Delivery	**Contractual**	M12	**Actual**	31-March-09
Status	version 1.0		final ☒	
Nature	prototype ☐ report ☒ dissemination ☐			
Dissemination Level	public ☒ consortium ☐			

Authors (Partner)	Florian Fischer (UIBK), Barry Bishop(UIBK)			
Resp. Author	Florian Fischer (UIBK)		**E-mail**	florian.fischer@sti2.at
	Partner	Partner	**Phone**	+43 512 507 6489

Abstract (for dissemination)	The minimal representation language for LarKC (see D1.3.1), called $L2$, deliberately has limited expressivity in order to meet its main objectives of scalability and tractability. The purpose of this deliverable is to perform an empirical analysis of these scalability properties on practical RDF repositories that not only store RDF triples, but also perform inference over them. By this means we attempt to gain insights into the feasibility of a practical implemention of $L2$ in addition to purely theoretical results. This should also help to analyse common performance pitfalls that exist even in very lightweight KR formalisms. In this deliverable we more closely investigate the sources of computational complexity for $L2$, first by briefly observing theoretical results and furthermore by a practical evaluation. For this purpose we identify the most important performance factors for practical inference on implementations that have to be considered for a useful performance analysis of $L2$. The deliverable documents the experimental setup parameters used for the evaluation of $L2$, such as the basic hardware and software setup and the chosen data-sets. Also included is a rational for the specific choices taken. Furthermore we give a brief overview of the concrete system as well as the configuration for the experiment. Finally, we provide the results of the analysis and the conclusions we can draw based on them, which should be helpful for the purpose of further refining $L2$ as well as for practical implementations.
Keywords	Benchmarking, Reasoning, Scalability

Version Log			
Issue Date	**Rev No.**	**Author**	**Change**
26.11.2008	1	Florian Fischer	Creation of Document
12.12.2008	2	Florian Fischer	Datasets & Introduction
5.03.2009	3	Florian Fischer	Version for internal review
10.03.2009	4	Vassil Momtchev	Peer Review
18.03.2009	5	Florian Fischer	Changes according to reviewer comments
19.03.2009	6	Florian Fischer	Final formatting

Project Consortium Information

Partner	Acronym	Contact
Semantic Technology Institute Innsbruck http://www.sti-innsbruck.at	STI	Prof. Dr. Dieter Fensel Semantic Technology Institute (STI) Innsbruck, Austria E-mail: dieter.fensel@sit2.at
AstraZeneca AB http://www.astrazeneca.com/	ASTRAZENECA	Bosse Andersson AstraZeneca Lund, Sweden E-mail: bo.h.andersson@astrazeneca.com
CEFRIEL SCRL. http://www.cefriel.it/	CEFRIEL	Emanuele Della Valle CEFRIEL SCRL. Milano, Italy E-mail: emanuele.dellavalle@cefriel.it
CYCORP, RAZISKOVANJE IN EKSPERIMENTALNI RAZVOJ D.O.O. http://cyceurope.com/	CYCORP	Michael Witbrock CYCORP, RAZISKOVANJE IN EKSPERIMENTALNI RAZVOJ D.O.O., Ljubljana, Slovenia E-mail: witbrock@cyc.com
Höchstleistungsrechenzentrum, Universität Stuttgart http://www.hlrs.de/	HLRS	Dr. Georgina Gallizo Höchstleistungsrechenzentrum, Universität Stuttgart Stuttgart, Germany E-mail : gallizo@hlrs.de
Max-Planck-Institut für Bildungsforschung http://www.mpib-berlin.mpg.de/index_js.en.htm	MAXPLANCKGESELLSCH	Dr. Lael Schooler, Max-Planck-Institut für Bildungsforschung Berlin, Germany E-mail: schooler@mpib-berlin.mpg.de
Ontotext Lab, Sirma Group Corp. http://www.ontotext.com/	ONTO	Atanas Kiryakov, Ontotext Lab, Sirma Group Corp. Sofia, Bulgaria E-mail: atanas.kiryakov@sirma.bg
SALTLUX INC. http://www.saltlux.com/EN/main.asp	Saltlux	Kono Kim SALTLUX INC Seoul, Korea E-mail: kono@saltlux.com
SIEMENS AKTIENGESELLSCHAFT http://www.siemens.de/	Siemens	Dr. Volker Tresp SIEMENS AKTIENGESELLSCHAFT München, Germany E-mail: volker.tresp@siemens.com
THE UNIVERSITY OF SHEFFIELD http://www.shef.ac.uk/	Sheffield	Prof. Dr. Hamish Cunningham THE UNIVERSITY OF SHEFFIELD Sheffield, UK E-mail: h.cunningham@dcs.shef.ac.uk
VRIJE UNIVERSITEIT AMSTERDAM http://www.vu.nl/	Amsterdam	Prof. Dr. Frank van Harmelen VRIJE UNIVERSITEIT AMSTERDAM Amsterdam, Netherlands E-mail: Frank.van.Harmelen@cs.vu.nl
THE INTERNATIONAL WIC INSTITUTE, BEIJING UNIVERSITY OF TECHNOLOGY http://www.iwici.org/	WICI	Prof. Dr. Ning Zhong THE INTERNATIONAL WIC INSTITUTE Mabeshi, Japan E-mail: zhong@maebashi-it.ac.jp

INTERNATIONAL AGENCY FOR RESEARCH ON CANCER http://www.iarc.fr/	IARC2	Dr. Paul Brennan INTERNATIONAL AGENCY FOR RESEARCH ON CANCER Lyon, France E-mail: brennan@iarc.fr

Table of Contents

1 Introduction and Motivation

A lot of research is being carried out to advance very expressive formalisms that add increasingly complex modelling constructs. However this increase in language expressivity is often intrinsically tied to higher computational cost as well. The resulting formalisms often have high theoretical complexity and are also difficult to implement efficiently in practice. As such, while these research results are valueable and produce important insights they are also are not directly applicable for knowledge representation and reasoning at a Web scale. A basic insight in this regard is that often already limited expressivity or rather a controlled level of language expressivity is sufficent to support many practical scenarios and furthermore that scalable implementations are possible for such lightweight languages. Thus it is also the approach adopted by many existing large scale inference systems and RDF repositories, which often only support a specific subset of standards such as OWL [6]. For example OWLIM[1], Oracle 11g[2], AllegroGraph RDFStore[3], Jena[4], OpenLink Virtuoso[5], and others all support only a certain incomplete subset of inferences sanctioned by RDF(S) and OWL. These restrictions are usually pragmatically oriented in order to (i) support "interesting" inferences, which are relevant to user requirements, and (ii) efficiently implementable.

In turn, *L*2, LarKC's minimal representation language (see D1.3.1), on purpose only supports a very basic set of inferences over RDF data. This deliverable aims to explore the scalability of inference defined by *L*2, with an emphasize on evaluating it on real implementations. For this purpose we identified suitable systems (RDF repositories) that support the intended level of inference or can be customized in this regard. We then proceeded to select data-sets, which are complex enough to also make clear use of the expressivity supported, and are thus challenging not only in terms of size but also in terms of reasoning required. The underlying motivation for this is that the main goal of this deliverable is not to perform an extensive query benchmark and to evaluate an individual system but rather to get a clear impression of the *inference* performance for *L*2 in different implementations in order to determine whether it is roughly in line with theoretical expectations.

For this purpose we briefly identify the relevant tasks an RDF repository has to deal with and the associated performance factors. We then execute performance tests that require different levels of reasoning, i.e. we deliberately perform benchmarks that delivered only incomplete results sets, in order to estimate (i) the associated performance gain, and (ii) the amount of information lost. In order to obtain results which can be generalized as much as possible we try to not emphasize specific optimizations made by a particular system, i.e. optimizations regarding the transitive closure of class and property hierarchies or individual equivalance. However we do acknowledge and point out these possible optimizations where applicable.

The rest of this document is structured as follows: Section 2 introduces the relevant background for this deliverable. It examines basic theoretical considerations for *L*2 which impact reasoning performance, as well as relevant tasks and performance metrics for RDF repository implementations. Section 3 describesthe experimental setup and

[1] `http://www.ontotext.com/owlim/index.html`
[2] `http://www.oracle.com/technology/tech/semantic_technologies/index.html`
[3] `http://agraph.franz.com/allegrograph/`
[4] `http://jena.sourceforge.net/`
[5] `http://virtuoso.openlinksw.com/`

data sets used in our evaluation. Consecutively, Section 4 introduces the system we used in our evaluation in more depth, including the setup used. Section 5 presents and in turn interprets the results of our evaluation. Finally, Section 6 concludes and summarizes this document.

2 Background

2.1 Theoretical Background of $L2$ and Related Work

In this section we briefly examine the theoretical properties of $L2$, and related languages. As described in D1.1.3, $L2$ is defined in terms of entailment rules (essentially Horn rules), which operate directly on RDF triples, similar to [13]. The restrictions placed on the language allow to for the decidability of ground entailment in polynomial time, while still retaining useful expressivity. There have been several approaches aiming at the same result, both in theory and in implementations.

In particular [12] elaborates on the possible time and space tradeoffs in RDF schema. The authors essentially discuss the implications of materialization of parts of the deductive closure of an RDF graph, which reduces the complexity of query answering, but increases the space requirements as well as maintenance overhead, e.g. in the case of updates. The paper describes experiments that identify certain modelling primitives that are the major source of this increase and in turn presents a technique that, based on the findings in the evaluation, computes only special parts of the deductive closure in order to avoid exactly these primitives. The purpose of this is to balance query time versus the space requirements of inference.

In the experiments multiple real world data sets, e.g. the CIA World Factbook, Wordnet, were used on a forward chaining system in order to to estimate the impact of materialization. The main results of the experiments can be summarized as follows: First of all, the particular expressivity of the underlying schema has a big impact on reasoning performance and space requirements. For relatively inexpressive schemata, such as the CIA World Factbook, standard RDFS reasoning mainly increases the amount of type statements in the closure. For more complex schemas, such as Wordnet, the increase of the subclass hierarchy has an increasingly noticeable impact as well. Consequently the authors suggest that these two focal points are main targets for optimization, with type inference being the preferred one. The reasons for this suggestion are that computing the type inference is a burden in combination with axiomatic triples, regardless of the complexity of the underlying schema. Furthermore, computing the transitive closure of the subclass hierarchy still pays off, as (i) it makes later inference concerning type statements almost trivial and (ii) the set of subclass statements in the schema can be considered to be more stable than instance related type information with respect to frequently changing data.

As such, these results are directly applicable to reasoning within $L2$ and provide valuable information for practical implementations. They are also already reflected in the design of $L2$ by the fact that a lot of type related inferences are already surpressed due to the exclusion of axiomatic triples. Moreover, [9] presents an in-depth theoretical discussion of the issues presented above, with the main focus on the complexity of entailment in various fragments of RDFS. It presents a more lightweight version of RDFS with favorable complexity properties as well, and underlines the theoretical feasibility of such an approach.

In a similar vain [4] examines the challenges of large scale reasoning with Web data by a rule-based approach using forward chaining as well. The authors also performed experimental benchmarking of their approach, which basically consists of a set of lightweight inferences, again similar to [13]. To achieve scalability they additionally performed a separation of schema statements and instance related statements. The authors point out the problem of including axiomatic triples and extensive schema reasoning, that adds a lot of information which is hardly used in the terminological part of a knowledge base. Thus they deliberately focus on performing inference over instance data. This is based on the observation that the amount of schema information is relatively small when compared to actual instance data, but that nearly all of the standard inference rules for instance data take schema information into account. Beyond that, [4] also discusses the difficulties and possible optimizations for equality reasoning, i.e. employing an in-memory index to keep track of individual equivalence. Again these are insights which directly translate to $L2$ implementations and are already partly considered in its definition.

For the specific rule-set of $L2$ itself we briefly recap the individual rules and additionally give relevant complexity measures per rule in Table 2.1 (listing only inferences that go beyond basic RDFS).

Rule (1) and (2) cover symmetry and transitivity of properties. Rules (3a) and (3b) formalize the notion that an individual can be considered to be equal to itself. Rule (4) captures reflexivity and rule (5) transitivity of individual equivalence. Rule (6) and (7) cover the semantics of inverse properties, including its reflexivity. Rules (8) and (9) denote that individuals that are classes or properties are considered sub-classes or sub-properties of themselves. These rules are important to facilitate basic meta-modelling in the language. Rule (10) denotes that existing relations are preserved when renaming nodes. Rules (11a), (11b) and (11c) express the semantics of class equivalence, while (12a), (12b) and (12c) do the same for property equivalence. Table 2.1 includes for each rule, the time complexity $\mathcal{T}$ for detecting a required rule application and the space complexity Δ for the number of triples inferred (the number of statements added to the deductive closure for an individual rule application).

As can be seen, the highest time complexity for an individual rule application in $L2$ is $O(n^3)$. The most complex rule covers transitive properties (Rule 2), which poses the same challenges as existing RDFS vocabulary, i.e. through transitive class hierarchies. As outlined above (see [12]) this is also one of the most promising optimization points. As a practical solution, the application of this rule on a graph can be mapped to a well studied problem, graph reachability, where efficient optimization algorithms exist see [11] [10] [14].

In the following, we outline the performance metrics that we consider useful in our practical evaluation. The main aim of the evaluation is to examine the reasoning performance of $L2$ in practice when compared to other similar fragments, which are readily implemented in existing systems, in order to study the tradeoff outlined before.

2.2 Performance Metrics

There are two very basic requirements for a large scale RDF repository/reasoning system. First of all, faced with very large data sets, it has to support the required efficiency and scalability. Secondly, it has to be able to support a powerful enough inference to show a clear benefit to the end users. These two requirements are usually at odds with each other and have to be carefully balanced. Furthermore, many systems only cover a specific subset of the existing standards, which makes a direct comparison between tools only based on scalability/raw performance very difficult, as it neglects a very basic metric — the complexity of reasoning involved. The bottom line is that it is important to consider the tradeoff between scalability and reasoning capabilities.

Rule No.	IF	THEN	$\mathcal{T}$	Δ
1	?p type SymmetricProperty ?v ?p ?w	?w ?p ?v	$O(n^2)$	$O(n)$
2	?p type TransitiveProperty ?u ?p ?v ?v ?p ?w	?u ?p ?w	$O(n^3)$	$O(n^2)$
3a	?v ?p ?w	?v sameAs ?v	$O(n)$	$O(n)$
3b	?v ?p ?w	?w sameAs ?w	$O(n)$	$O(n)$
4	?v sameAs ?w	?w sameAs ?v	$O(n)$	$O(n)$
5	?u sameAs ?v ?v sameAs ?w	?u sameAs ?w	$O(n^2)$	$O(n^2)$
6	?p inverseOf ?q ?v ?p ?w	?w ?q ?v	$O(n^2)$	$O(n)$
7	?p inverseOf ?q ?v ?q ?w	?w ?p ?v	$O(n^2)$	$O(n)$
8	?v type Class ?v sameAs ?w	?v subClassOf ?w	$O(n^2)$	$O(n)$
9	?p type Property ?p sameAs ?q	?p subPropertyOf ?q	$O(n^2)$	$O(n)$
10	?u ?p ?v ?u sameAs ?w ?v sameAs ?q	?w ?p ?q	$O(n^3)$	$O(n)$
11a	?v equivalentClass ?w	?v subClassOf ?w	$O(n)$	$O(n)$
11b	?v equivalentClass ?w	?w subClassOf ?v	$O(n)$	$O(n)$
11c	?v subClassOf ?w ?w subClassOf ?v	?v equivalentClass ?w	$O(n^2)$	$O(n)$
12a	?v equivalentProperty ?w	?v subProperty ?w	$O(n)$	$O(n)$
12b	?v equivalentProperty ?w	?w subProperty ?v	$O(n)$	$O(n)$
12c	?v subPropertyOf ?w ?w subPropertyOf ?v	?v equivalentProperty ?w	$O(n^2)$	$O(n)$

Table 2.1: Intended semantics for $L2$ given by means of first-order implications / entailment rules.

So for our practical evaluation we focused on measuring this tradeoff and in turn identified several relevant tasks and metrics which have to be considered. Several of the metrics we identified are standard metrics, i.e. they apply also for database benchmarks, namely load time, query response time, and data size. However, since we are primarily interested in the reasoning performance, we have to extend these measures slightly to take this additional complexity into account. We do this by performing tests at different degrees of inference and by taking query completeness into account.

Load Time

[15] identifies two main stages in ontology processing: Loading and pre-processing the data and query answering. *Load time* refers to the stand-alone time elapsed for storing test data in a system, as in a traditional database benchmark. Beyond that the load time also includes the time spent for processing the test data, which includes parsing and might also include reasoning, such as in the case when inference is performed immediately by forward chaining. This highlights the aforementioned tradeoff, because pre-computing implicit statements and indexing usually implies longer load time, but improves query performance. A concrete metric for the loading performance can be defined as time required to load a specific number of statements:

$$\text{Loading speed} = \text{Number of statements} \; / \; \text{Time}$$

Query Response Time

The second main step in ontology processing is concerned with evaluating queries over a given data set. The first aspect involved in this is *query response time*, which can be measured similarly as in database systems. For example, the Berlin SPARQL Benchmark[1] is a benchmark that very much focuses on query evaluation and processing time, and less on the inference aspect and expressivity of the underlying schema.

Processing a query usually involves several distinct steps as well: A query needs to be parsed, validated and suitably decomposed in order to generate an execution plan. It is then evaluated and the results are returned. Contrary to inferences done by means of forward chaining and materializing implicit information at loading time, a query can be backward chained deductively.

Various factors can influence query response time, such as the size and complexity of the data set under consideration, the amount of results to be returned, and last, but not least, the complexity of the query itself (i.e. LIKE clauses or the number of triple patterns involved). Furthermore, in order to allow query optimizations and caching to reflect practical behavior it is usual to allow systems to reach a steady state by several warm-up queries before realistic results can be achieved.

Completeness of Query Results

Completeness of the query results is a factor that goes beyond the metrics needed for traditional DBMS systems. Completeness as traditionally considered means that an inference system can conclude all statements from a knowledge base that are logically

[1] `http://www4.wiwiss.fu-berlin.de/bizer/BerlinSPARQLBenchmark/spec/index.html`

logically entailed by it. As mentioned, the amount of reasoning performed by practical systems (and thus also the kind of queries that they can answer) often varies by a great deal. While incomplete systems (with respect to a standard such as OWL) are still useful due to the performance benefits they might offer, it is important consider completeness of query results as a metric.

Size and Complexity of the Data Set

This metric covers two distinct aspects concerned with the data set used: The first aspect is concerned with the initial size of the data set and the size of the data set after actually loading it in to the system. These two numbers are typically quite different depending on the amount of preprocessing and reasoning performed by the particular system as well as the second important aspect that has to be considered, namely the complexity of the data set itself. In this sense, highly connected data sets, i.e. featuring long transitive chains or also very deep class hierarchies, increase the complexity of reasoning and in turn also the number of statements the system has to deal with after initially loading it. The complexity of individual rules in $L2$ can be considered as a basic example to illustrate this point.

3 Datasets and Queries

In this deliverable we want to focus on the complexity of reasoning and the degree of inferences permitted by the lightweight language $L2$, as defined in D1.1.3. As such the emphasis is on reasoning performance and completeness of specific "entailment regimes" and not strictly on query answering in isolation. For this reason, careful consideration in the selection of data-sets is required. A suitable data-set should:

1. Require a considerable amount of reasoning in order to highlight both the amount of information which is lost if reasoning is restricted, but also the trade off in terms of computational performance to be made.

2. Reflect a "typical" data-set in the sense that results allow for the drawing of conclusions for a broader range of scenarios.

With this in mind, one established synthetic benchmark was selected, the Lehigh University Benchmark, which focuses on moderately complex, data driven reasoning, and a real word data set in the form of the RDF/OWL representation of Wordnet. This allows us to operate in a replicable, well defined scenario, as well as in a typical application setting in order to get a more comprehensive picture of the behavior of $L2$.

3.1 LUBM

The Lehigh University Benchmark (LUBM) [2] is a synthetic benchmark to measure the scalability of OWL repositories. Benchmark data-sets are synthetically generated according to specific parameters. They consist of instance data over a fixed OWL ontology describing a university organization. In this fashion it is possible to facilitate repeatable experiments, which can be scaled to an arbitrary size.

The schema used is of moderate complexity and size (in contrast to many other DL benchmarks). Technically it is expressed within the OWL-Lite fragment and defines 43 classes and 32 properties (including 25 object properties and 7 data-type properties). The ontology uses the following language features from OWL-Lite: `inverseOf`, `TransitiveProperty`, `someValuesFrom`, `intersectionOf`. This level of complexity roughly corresponds to the expressivity provided by [13].

This composition of the benchmark is based on the assumption that the size of instance data (assertions) will outweigh the size of the schema, as understood in the context of the Semantic Web. In this sense LUBM allows for increasing the number of assertional statements in order to mimic predominantly data driven reasoning along with extensional queries concerned with instance data.

For this purpose the LUBM benchmark also includes 14 queries, which are used to check the query evaluation and the correctness/completeness of the results. Thus we can determine the trade-off between reasoning complexity and scalability of inference for this evaluation scenario.

The biggest standard data set is LUBM(50,0), which contains generated data for 50 universities. This data set contains roughly 6.89 million triples as explicit statements, which entail a total of roughly 11 million statements. In order to get a better impression regarding the scalability towards bigger data sets, we included more challenging scenarios in the form of LUBM(200,0) (around 28 million explicit statements and 45 million implicit statements), and LUBM(500,0) (roughly 68 million explicit statements and 112 million implicit statements).

3.2 Wordnet RDF/OWL

WordNet [7] is a lexical database for the English language, also available as OWL ontology. Wordnet's OWL version is comparatively small with only 1.9 million triples. However Wordnet has a considerably more expressive schema than LUBM and expands to roughly 8.2 million triples when fully materialized.

Wordnet's schema is centered around three primary classes: `Synset`, `WordSense` and `Word`. `Synset` and `Wordsense` are base classes for distinct lexical groups, e.g. `VerbSynset`, `VerbWordSense`. There are no subclasses of `Word` since it represents a simple word without any precise association to its lexical function.

There are primarily three main uses of properties in Wordnet: First of all to link the three main classes, i.e. by connecting a Synset with its WordSenses and in turn its WordSenses with concrete Words. Furthermore, properties are used to model relations such as hyponyms, for example in order to relate synsets to each other.

Considering language features, Wordnet makes use of basic RDFS and the following features from OWL in order to provide additional semantics if desired:
`owl:disjointWith` (to form statements between classes such as Word and WordSense), `owl:allValuesFrom`, `owl:someValuesFrom` (both in order to define additional restrictions), `owl:TransitiveProperty`, `owl:inverseOf` (for example to state that hypernymOf is an inverse of hyponymOf). While the first extensions only provide the possibility to check consistency of the data-set, the last two constructs from OWL are essential since a fair amount of information is not explicitly available in the data-set, but requires reasoning.

Typically, queries would consider asking for all synonyms having the same meaning as some concept, all hypernyms that are more general than a certain concept, etc. A peculiarity of the data-set is the transitivity chains formed and also required for answering such queries properly, which in turn requires a considerable amount of reasoning.

Following we now give a brief overview of OWLIM and the configuration of the system on which we deployed these data-sets.

4 Description of Test Environment

In this section we briefly describe the system setup we used in our evaluation. We first give an overview of OWLIM, which we used in our experiments, and motivate why it was chosen to conduct our tests. Then we lay out other underlying aspects of the test environment, such as hardware and software configuration.

4.1 Owlim

Since this deliverable is first and foremost concerned with the reasoning performance of $L2$ in relation to RDFS and more extensive OWL inference an evaluation systems obviously needs to be able to (i) perform a significant amount of inference and (ii) make it possible to tailor the inference specifically to the semantics of $L2$ as well. Thus it is not possible to use systems that only perform only very limited amounts of reasoning, for which the precise kind of inference performed is not clearly defined, or which are not extensible in the previously mentioned fashion.

In turn we conducted our evaluations on BigOWLIM [5] using several standard configurations supplied in addition to a custom tailored configuration for $L2$ in order to compare performance as well as the amount of reasoning performed in relation to that across several different scenarios.

OWLIM is a semantic repository, implemented in Java, which uses TRREE, a specialized rule engine in order to perform RDF and (partial) OWL inference. OWLIM can be used as a Storage and Inference Layer for the Sesame framework and supports several distinct pre-defined rule-sets, which configure the desired level of semantics. Additionally OWLIM provides the possibility to integrate custo,m user-defined rule sets, which makes it a suitable platform for our evaluation of $L2$'s performance.

As basic reasoning strategy OWLIM uses forward chaining according to the configured rule set, with all the associated advantages and disadvantages. Uploading, reasoning and query evaluation are fast, but on the other hand deletion of statements is a relatively costly operation. Furthermore, the amount of implicit statements can grow very fast and include many redundant statements in the end, as outlined in Section 2.

As mentioned OWLIM is distributed with different preconfiguredsets of entailment rules called OWL-Max, OWL-Horst, and RDFS. RDFS provides the usual RDFS semantics as defined in [3] but skips parts of the data type reasoning due to performance reasons. OWL-Horst (the default rule set) corresponds roughly to the set of pD* entailment rules from [13]. While OWL-Horst basically covers the same expressivity as DLP [1] it does not constrain the RDFS semantics and is very close to the upcoming OWL 2 RL profile [8]. OWL-Max, as the most comprehensive semantics in OWLIM, is close to OWL-Horst but goes slightly beyond it (its expressiveness covers OWL Lite combined with unrestricted RDFS).

In addition to these different semantic levels OWLIM in addition supports optional optimizations to suppress a set of inefficient axioms and rules which typically only lead to trivial and not very relevant inferences, mostly related to inferences based on RDFS' axiomatic triples and basic type information. While these optimizations affect each of the predefined rule sets, $L2$ adopts these optimizations by default, as do several other related approaches (see Section 2). We do however consider these optional optimizations in our evaluation where applicable.

OS	Debian Etch, Kernel 2.6.24-22-generic
BigOWLIM	Version 3.1.a2
Java	Version "1.6.0_07" Java HotSpot(TM) Sever VM (build 10.0-b23, mixed mode) Max. Heap Size 1.5 GB
CPU Model	Quad-Core AMD Opteron(tm) Processor 2352
CPU Speed	2100 MHz
Cache Size	512 KB
MemTotal	2062320 kB
MemFree	1772800 kB
SwapTotal	3903752 kB
SwapFree	3903752 kB

Table 4.1: Summary Hard- and Software setup.

For the sake of brevity we do not give the complete definition of the custom $L2$ rules here, but rather list the complete configuration used in Appendix A on page 19. It should be noted that those rules have *not* been over-proportionally optimized as this is typically also not possible without an in-depth understanding of implementation details.

4.2 Test Environment

Following we describe the test environment employed in our evaluation in terms of the used underlying hard- and software in more detail. We performed all our test runs on a single, identical machine. The test machine was a commodity, off-the shelf Quad-Core Opteron server running Debian Etch.

The machine had an AMD Opteron 2352 processor, with 4 cores at 2,1GHz and 512kb each for each of them. In terms of memory the machine was equipped with 2GB of RAM and 200GB of hard disk. Of the main memory 1.5 GB where set as maximum heap space for Java. Concerning the software used, we employed a Java server VM, version 1.6.0_07 along with BigOWLIM version 3.1.a2. The overall configuration is summarized in Table 4.1.

5 Results

In this section we consequently present our evaluation results obtained in regard to the previously defined metricss and with the outlined experimental setup. We mainly focus on scalability of inference and also the amount of results returned instead of query performance. First we present the results obtained in multiple runs using the LUBM data sets, followed by results in which Wordnet was used. We present results in a condensed form in this section. The complete output of the evaluation runs can be found in Appendix B on page 22.

Initially we conducted a series of detailed experiments to establish the behavior of the several fragments relative to each other. For this purpose we used LUBM(50,0) as test data set, and more particularly we tried to establish the completeness of the different inference levels. The results of these initial tests are summarized in Table 5.1. In general OWL-Horst (and in turn of course also OWL-Max) is sufficient to reaches completeness for the 14 reference queries included in LUBM, while neither general RDFS reasoning nor $L2$ are sufficiently expressive to derive all the desired inferences. The reason for this is that both $RDFS$ and $L2$ lack specific language constructs or rather a combination of them. However, $L2$ derives more implicit statements than even OWL-Max, which can be explained by the absence of specific Description Logic language constructs used to enforce restrictions. A noteworthy result is that despite the fact that $L2$ ends up with the largest amount of implicitly derived statements, OWLIM is still able to load the complete data set faster for $L2$ than for any of the OWL inference variants, except OWL-Horst with enabled optimizations. This in turn results in a higher processing speed per statement for $L2$, apart from a comparison to plainly loading the data set with no inference at all and for RDFS reasoning.

Inference	**Load time (sec.)**	**Total No. of Statements**	**No. Of Explicit Statements**	**Load Speed (st. / sec.)**
None	157	6655015	6655015	42388,63
L2	472	12371580	6655017	26210,97
RDFS	244	9401114	6655065	38529,16
RDFS + Opt.	203	8318054	6655046	40975,64
OWL Horst	506	12087215	6655074	23887,78
OWL Horst + Opt.	433	11004153	6655055	25413,75
OWL Max	622	12232774	6655074	19666,84
OWL Max + Opt.	523	11149715	6655055	21318,77

Table 5.1: LUBM(50,0) results

We then proceeded to increase to size of the data set to LUBM(200,0) and respectively LUBM(500,0). In these cases we only examined the raw loading speed, and inference based on the $L2$ rule set and OWL-Max, the most expressive variants supported by OWLIM (albeit with optimizations enabled). The respective results are recorded in Tables 5.2 and 5.3. In both cases reasoning develops an increasing impact on total loading time. However, $L2$ can again be processed faster than OWL-Max, both in terms of load time as well as in terms of triples processed per second. This further substantiates that the entailment rules of $L2$ are inherently less complex as i.e. those of OWL-Horst or OWL-Max (even when including optimizations).

Inference	**Load time (sec.)**	**Total No. of Statements**	**No. Of Explicit Statements**	**Load Speed (st. / sec.)**
None	706	26697031	26697031	37814,49
L2	2208	49616662	26697033	22471,31
OWL Max + Opt.	2331	44725123	26697071	19187,1

Table 5.2: LUBM(200,0) results

Inference	**Load time (sec.)**	**Total No. of Statements**	**No. Of Explicit Statements**	**Load Speed (st. / sec.)**
None	2006	66731653	66731653	33266,03
L2	7920	124005498	66731655	15657,26
OWL Max + Opt.	8183	111799180	66731693	13662,37

Table 5.3: LUBM(500,0) results

Furthermore we also collected results using Wordnet as an example of a distinctly more complex data set. The results of these measurements are depicted in Table 5.4. The most obvious general observation is that the total load time as well as the number of derived statements increase much faster as in the synthetic scenarios of LUBM. Based on the complex underlying schema, OWL-Max makes it possible to derive more implicit statements than $L2$ in this scenario, however it requires roughly twice the time for this purpose.

This trade off for the various variants (both including and excluding optimizations wherever applicable), is shown in Figure 5.1 for LUBM and in Figure 5.2 for Wordnet.

What can be seen in these plots is that in the case of LUBM the $L2$ rule set performs similarly well, but it also highlights its weakness: Although $L2$ is able to perform a considerable amount of the required inference it is not able to answer all of the reference queries correctly, because it lacks certain specific language constructs, i.e. universal quantification and others. On the other hand, of all the predefined fragments only the optimized OWL-Horst rule set results in faster evaluation.

In Figure 5.2 the increased expressivity of OWL-Max in combination with a data set that actually uses these modelling constructs becomes apparent, both in terms of implicit statements derived (Wordnet requires OWL-Max for complete inference) as well as in terms of reasoning time. Moreover, for Wordnet OWLIM takes considerable time to materialize the data, despite its relatively modest size, even when only using RDFS entailment.

Inference	Load time (sec.)	Total No. of Statements	No. Of Explicit Statements	Load Speed (st. / sec.)
None	48	1943045	1943045	841810
RDFS + Optimization	98	4571025	1943075	841821
RDFS	120	5036048	1943094	841821
L2	304	7558124	1943047	841810
OWL Horst	259	6163339	1943084	841826
OWL Horst + Opt.	298	6628353	1943103	841826
OWL Max	637	8641440	1943103	841838
OWL Max + Opt.	566	8176412	1943084	841838

Table 5.4: Wordnet results

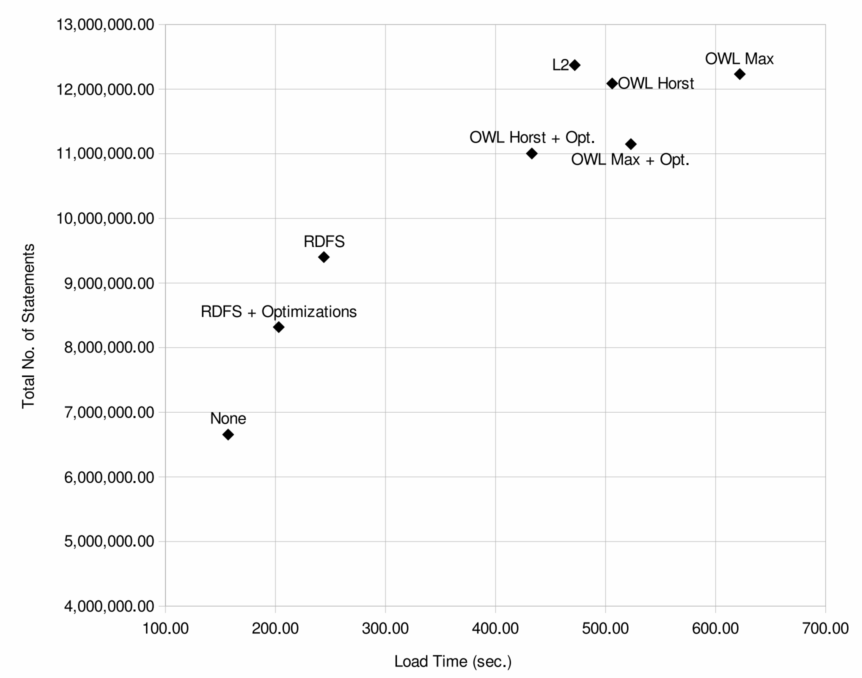

Figure 5.1: Total number of statements after materialization of LUBM(50,0) compared to load time.

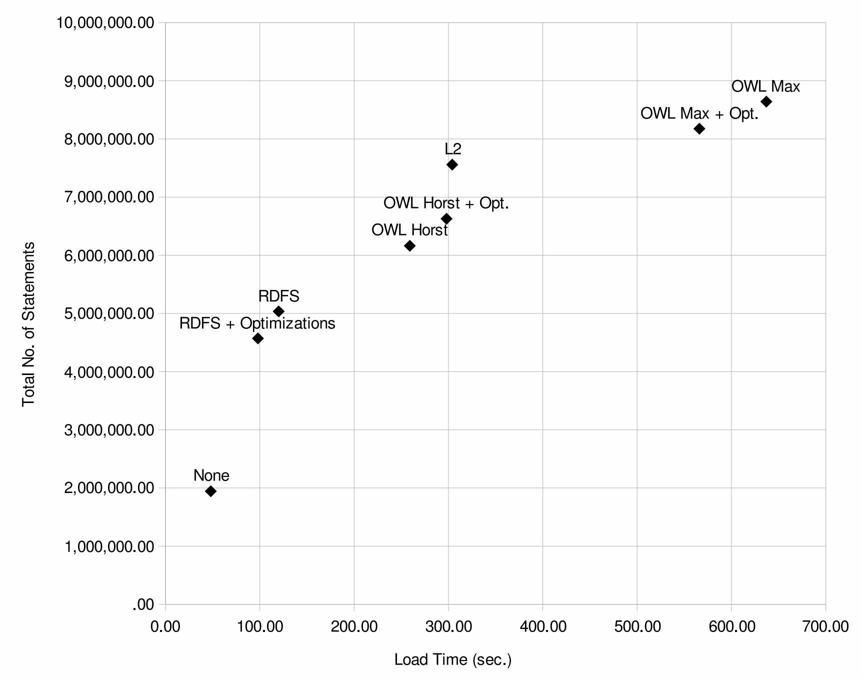

Figure 5.2: Total number of statements after materialization of Wordnet compared to load time.

6 Conclusion

In this deliverable we presented an initial empirical evaluation of $L2$, the lightweight set of entailment rules presented in D1.1.3. For this purpose we first examined related approaches and theoretical considerations. We then proceeded with a practical evaluation using OWLIM and a custom rule set in order to compare it against predefined configurations. In most cases $L2$ performed comparable to highly tuned and optimized rule sets. The rather simple schema of the LUBM data set slightly favors $L2$ in comparison to OWL-Max. Beyond this, the absence of specific constructs from $L2$ became apparent in the evaluation, and it remains to consider if an extension towards them is worthwhile.

Several other relevant observations can be drawn from the results of this deliverable. First of all, existing work on optimizations of reasoning systems which perform very lightweight OWL inference or RDFS based inference often directly carry over to $L2$ and obviously also to LarKC as a whole. These approaches serve as very basic and pragmatic approximations and should be taken into account during the development of implementations. Furthermore it became apparent that for a more detailed study especially a finer grained metric of completeness is needed, i.e. the actual relevance of an inference results should ideally be taken into account i.e. as a weighting factor. Again this is potentially valuable future work.

References

[1] B. Grosof, I. Horrocks, R. Volz, and S. Decker. Description Logic Programs: Combining Logic Programs with Description Logic. 2003.

[2] Y. Guo, Z. Pan, and J. Heflin. LUBM: A benchmark for OWL knowledge base systems. *Web Semantics: Science, Services and Agents on the World Wide Web*, 3(2-3):158–182, 2005.

[3] P. Hayes and B. McBride. RDF Semantics. *W3C Recommendation*, 10, 2004.

[4] Aidan Hogan, Andreas Harth, and Axel Polleres. Saor: Authoritative reasoning for the web. In *ASWC '08: Proceedings of the 3rd Asian Semantic Web Conference on The Semantic Web*, pages 76–90, Berlin, Heidelberg, 2008. Springer-Verlag.

[5] A. Kiryakov, D. Ognyanov, and D. Manov. Owlim-a pragmatic semantic repository for owl. *Lecture Notes in Computer Science*, 3807:182, 2005.

[6] D.L. McGuinness, F. van Harmelen, et al. OWL Web Ontology Language Overview. *W3C Recommendation*, 10:2004–03, 2004.

[7] G.A. Miller. WordNet: a lexical database for English. *Communications of the ACM*, 38(11):39–41, 1995.

[8] B. Motik and I. Horrocks. OWL 2 Web Ontology Language. *W3C Working Draft, W3C*, 2008.

[9] S. Munoz, J. Perez, and C. Gutierrez. Minimal Deductive Systems for RDF. *LECTURE NOTES IN COMPUTER SCIENCE*, 4519:53, 2007.

[10] R. Schenkel, A. Theobald, and G. Weikum. HOPI: An Efficient Connection Index for Complex XML Document Collections. *Lecture Notes in Computer Science*, pages 237–255, 2004.

[11] R. Schenkel, A. Theobald, and G. Weikum. Efficient Creation and Incremental Maintenance of the HOPI Index for Complex XML Document Collections. In *Proceedings of the International Conference on Data Engineering*, volume 21, page 360. IEEE Computer Society Press; 1998, 2005.

[12] H. Stuckenschmidt and J. Broekstra. Time-space trade-offs in scaling up rdf schema reasoning. *Lecture notes in computer science*, 3807:172, 2005.

[13] Herman J. ter Horst. Combining rdf and part of owl with rules: Semantics, decidability, complexity. In Yolanda Gil, Enrico Motta, V. Richard Benjamins, and Mark A. Musen, editors, *International Semantic Web Conference*, volume 3729 of *Lecture Notes in Computer Science*, pages 668–684. Springer, 2005.

[14] H. Wang, H. He, J. Yang, P.S. Yu, and J.X. Yu. Dual labeling: Answering graph reachability queries in constant time. In *Proceedings of the 22nd International Conference on Data Engineering (ICDE)*, page 75, 2006.

[15] T. Weithoner, T. Liebig, M. Luther, and S. Bohm. What's Wrong with OWL Benchmarks? In *Proceedings of the Second International Workshop on Scalable Semantic Web Knowledge Base Systems (SSWS 2006)*, pages 101–114, 2006.

A Appendix I — L2 Rule Set

Following we concisely list the custom rule set for $L2$, used as configuration in OWLIM. Note that this ruleset is a direct implementation of $L2$'s inference rules, and does not take any specific possible optimizations into account at this point. The only relevant change is that in certain cases rules have been concatenated for convenience, as for example `rdf1`, `rdfs4a`, and `rdfs4b`. Others have been split up, i.e. `rdfp11`.

```
Prefices
{
     rdf      :  http://www.w3.org/1999/02/22-rdf-syntax-ns#
     rdfs     :  http://www.w3.org/2000/01/rdf-schema#
     owl      :  http://www.w3.org/2002/07/owl#
     xsd      :  http://www.w3.org/2001/XMLSchema#
}

Axioms
{
// All axiomatix triples excluded.
// The following are two trivially true statements as required by Owlim.
<rdfs:subClassOf>  <rdf:type>  <owl:TransitiveProperty>
<rdfs:subPropertyOf>  <rdf:type>  <owl:TransitiveProperty>
}

Rules
{
// Simple type information for triples.
Id: rdf1_rdfs4a_4b
     x  a  y
    -------------------------------
     a  <rdf:type>  <rdf:Property>             [Constraint a != <rdf:type>]
     x  <rdf:type>  <rdfs:Resource>
     a  <rdf:type>  <rdfs:Resource>
     y  <rdf:type>  <rdfs:Resource>

//Domain and range standard rules.
Id: rdfs2
     x  a  y                                   [Constraint a != <rdf:type>]
     a  <rdfs:domain>  z                       [Constraint z != <rdfs:Resource>]
    -------------------------------
     x  <rdf:type>  z

Id: rdfs3
     x  a  u
     a  <rdfs:range>  z                        [Constraint z != <rdfs:Resource>]
    -------------------------------
     u  <rdf:type>  z

// Transitivity of rdfs:subPropertyOf.
Id: rdfs5
     a  <rdfs:subPropertyOf>  b                [Constraint a != b, a != c]
     b  <rdfs:subPropertyOf>  c                [Constraint b != c, a != b]
    -------------------------------
     a  <rdfs:subPropertyOf>  c

// Basic rules to support properties.
Id: rdfs6
    a  <rdf:type> <rdf:Property>
    -------------------------------
    a  <rdfs:subPropertyOf>  a

Id: rdfs7
     x  a  y
     a  <rdfs:subPropertyOf>  b                [Constraint a != b]
    -------------------------------
     x  b  y

// Basic rules to support classes.
Id: rdfs8_10
     x  <rdf:type>  <rdfs:Class>
```

```
    -------------------------------
     x  <rdfs:subClassOf>  <rdfs:Resource>
     x  <rdfs:subClassOf>  x

// Transitivtiy of rdfs:subClassOf
Id: rdfs9
     a  <rdf:type>  x
     x  <rdfs:subClassOf>  y                 [Constraint x != y]
    -------------------------------
     a  <rdf:type>  y

Id: rdfs11
     x  <rdfs:subClassOf>  y                 [Constraint x != y, x != z]
     y  <rdfs:subClassOf>  z                 [Constraint y != z, y != x]
    -------------------------------
     x  <rdfs:subClassOf>  z

// Containers and datatypes.
Id: rdfs12
     x  <rdf:type>  <rdfs:ContainerMembershipProperty>
    -------------------------------
     x  <rdfs:subPropertyOf>  <rdfs:member>

Id: rdfs13
     x  <rdf:type>  <rdfs:Datatype>
    -------------------------------
     x  <rdfs:subClassOf>  <rdfs:Literal>


//===============================================================
// OWL/RDFS++ Part
//===============================================================

Id: rdfp5
     x  p  y
    -------------------------------
     x  <owl:sameAs>  x

Id: rdfp6
     x  <owl:sameAs>  y
    -------------------------------
     y  <owl:sameAs>  x

// Support for inverse properties and their symmetric behavior.
Id: rdfp8ax
     x  p  y
     p  <owl:inverseOf>  q
    -------------------------------
     y  q  x

Id: rdfp8bx
     x  q  y
     p  <owl:inverseOf>  q
    -------------------------------
     y  p  x

// SymmetricProperty
Id: rdfp3
     p  <rdf:type>  <owl:SymmetricProperty>
     x p  y
    -------------------------------
     y p  x

// Transitivity
Id: rdfp4
     p  <rdf:type>  <owl:TransitiveProperty>
     x  p  y                                 [Constraint x != y]
     y  p  z                                 [Constraint y != z]
    -------------------------------
     x  p  z

Id: rdfp11a
     x  <owl:sameAs>  y                      [Constraint x != y]
```

```
     x  p  z
    -------------------------------
     y  p  k

Id: rdfp11b
     z  <owl:sameAs>  y                    [Constraint z != y]
     x  p  z
    -------------------------------
     x  p  y

Id: rdfp11c
     p  <owl:sameAs>  y                    [Constraint p != y]
     x  p  z
    -------------------------------
     x  y  k

//Equivalence

Id: rdfp7
     x  <owl:sameAs>  y                    [Constraint x != y]
     y  <owl:sameAs>  z                    [Constraint y != z]
    -------------------------------
     x  <owl:sameAs>  z

Id: rdfp10
     p  <owl:sameAs>  q
     p  <rdf:type> <rdf:Property>
    -------------------------------
     p  <rdfs:subPropertyOf>  q

Id: rdfp9
     p  <owl:sameAs>  q
     p  <rdf:type> <rdfs:Class>
    -------------------------------
     p  <rdfs:subClassOf>  q

//Property and Class equivalence
Id: rdfp13ab
     p  <owl:equivalentProperty>  q
    -------------------------------
     p  <rdfs:subPropertyOf>  q
     q  <rdfs:subPropertyOf>

Id: rdfp12ab
     p  <owl:equivalentClass>  q
    -------------------------------
     p  <rdfs:subClassOf>  q
     q  <rdfs:subClassOf>  p

Id: rdfp12c
     x  <rdfs:subClassOf>  y               [Constraint y != x]
     y  <rdfs:subClassOf>  x               [Cut]
    -------------------------------
     x  <owl:equivalentClass>  y

Id: rdfp13c
     x  <rdfs:subPropertyOf>  y            [Constraint y != x]
     y  <rdfs:subPropertyOf>  x            [Cut]
    -------------------------------
     x  <owl:equivalentProperty>  y
}
```

B Appendix II —- Complete Benchmark Results

B.1 LUBM Output

Following we list the complete result output of our test runs using the respective LUBM datasets. Output is organized according to the configured level of inference used during the run.

No Inference

```
Finished loading OWLIM_50 157 seconds

OWLIM_50 157 seconds

### Started testing OWLIM_50 ###

~~~query1~~~
Query optimized in 0 ms.
Owlim evaluation strategy : optimized in 0 ms.
Duration: 0 Result#: 4

~~~query2~~~
Query optimized in 46 ms.
Owlim evaluation strategy : optimized in 46 ms.
Duration: 734 Result#: 130

~~~query3~~~
Query optimized in 0 ms.
Owlim evaluation strategy : optimized in 0 ms.
Duration: 0 Result#: 6

~~~query4~~~
Query optimized in 0 ms.
Owlim evaluation strategy : optimized in 0 ms.
Duration: 0 Result#: 0

~~~query5~~~
Query optimized in 0 ms.
Owlim evaluation strategy : optimized in 0 ms.
Duration: 0 Result#: 0

~~~query6~~~
Query optimized in 0 ms.
Owlim evaluation strategy : optimized in 0 ms.
Duration: 0 Result#: 0

~~~query7~~~
Query optimized in 0 ms.
Owlim evaluation strategy : optimized in 0 ms.
Duration: 0 Result#: 0

~~~query8~~~
Query optimized in 0 ms.
Owlim evaluation strategy : optimized in 0 ms.
Duration: 0 Result#: 0

~~~query9~~~
Query optimized in 0 ms.
Owlim evaluation strategy : optimized in 0 ms.
Duration: 0 Result#: 0

~~~query10~~~
Query optimized in 0 ms.
Owlim evaluation strategy : optimized in 0 ms.
Duration: 16 Result#: 0

~~~query11~~~
Query optimized in 0 ms.
```

```
Owlim evaluation strategy : optimized in 0 ms.
Duration: 0 Result#: 0

~~~query12~~~
Query optimized in 0 ms.
Owlim evaluation strategy : optimized in 0 ms.
Duration: 0 Result#: 0

~~~query13~~~
Query optimized in 0 ms.
Owlim evaluation strategy : optimized in 0 ms.
Duration: 0 Result#: 0

~~~query14~~~
Query optimized in 0 ms.
Owlim evaluation strategy : optimized in 0 ms.
Duration: 406 Result#: 393730
### Finished testing OWLIM_50 ###

OWLIM_50
Time/Result#
query1 0 4
query2 734 130
query3 0 6
query4 0 0
query5 0 0
query6 0 0
query7 0 0
query8 0 0
query9 0 0
query10 16 0
query11 0 0
query12 0 0
query13 0 0
query14 406 393730
NumberOfStatements = 6655015
NumberOfExplicitStatements = 6655015
NumberOfEntities = 1639962

Finished loading OWLIM_200 706 seconds

OWLIM_200 706 seconds

### Started testing OWLIM_200 ###

~~~query1~~~
Query optimized in 16 ms.
Owlim evaluation strategy : optimized in 16 ms.
Duration: 31 Result#: 4

~~~query2~~~
Query optimized in 63 ms.
Owlim evaluation strategy : optimized in 63 ms.
Duration: 29719 Result#: 499

~~~query3~~~
Query optimized in 16 ms.
Owlim evaluation strategy : optimized in 31 ms.
Duration: 31 Result#: 6

~~~query4~~~
Query optimized in 16 ms.
Owlim evaluation strategy : optimized in 16 ms.
Duration: 16 Result#: 0

~~~query5~~~
Query optimized in 0 ms.
Owlim evaluation strategy : optimized in 0 ms.
Duration: 0 Result#: 0

~~~query6~~~
Query optimized in 0 ms.
Owlim evaluation strategy : optimized in 0 ms.
```

```
Duration: 0 Result#: 0

~~~query7~~~
Query optimized in 16 ms.
Owlim evaluation strategy : optimized in 16 ms.
Duration: 16 Result#: 0

~~~query8~~~
Query optimized in 0 ms.
Owlim evaluation strategy : optimized in 0 ms.
Duration: 15 Result#: 0

~~~query9~~~
Query optimized in 0 ms.
Owlim evaluation strategy : optimized in 0 ms.
Duration: 0 Result#: 0

~~~query10~~~
Query optimized in 0 ms.
Owlim evaluation strategy : optimized in 0 ms.
Duration: 0 Result#: 0

~~~query11~~~
Query optimized in 0 ms.
Owlim evaluation strategy : optimized in 0 ms.
Duration: 16 Result#: 0

~~~query12~~~
Query optimized in 0 ms.
Owlim evaluation strategy : optimized in 0 ms.
Duration: 0 Result#: 0

~~~query13~~~
Query optimized in 0 ms.
Owlim evaluation strategy : optimized in 0 ms.
Duration: 0 Result#: 0

~~~query14~~~
Query optimized in 0 ms.
Owlim evaluation strategy : optimized in 0 ms.
Duration: 28094 Result#: 1584743
### Finished testing OWLIM_200 ###

OWLIM_200
Time/Result#
query1 31 4
query2 29719 499
query3 31 6
query4 16 0
query5 0 0
query6 0 0
query7 16 0
query8 15 0
query9 0 0
query10 0 0
query11 16 0
query12 0 0
query13 0 0
query14 28094 1584743
NumberOfStatements = 26697031
NumberOfExplicitStatements = 26697031
NumberOfEntities = 6575127

Finished loading OWLIM_500 2006 seconds

OWLIM_500 2006 seconds

### Started testing OWLIM_500 ###

~~~query1~~~
Query optimized in 16 ms.
Owlim evaluation strategy : optimized in 16 ms.
Duration: 16 Result#: 4
```

```
~~~query2~~~
Query optimized in 31 ms.
Owlim evaluation strategy : optimized in 31 ms.
Duration: 84093 Result#: 1284

~~~query3~~~
Query optimized in 32 ms.
Owlim evaluation strategy : optimized in 32 ms.
Duration: 32 Result#: 6

~~~query4~~~
Query optimized in 15 ms.
Owlim evaluation strategy : optimized in 15 ms.
Duration: 15 Result#: 0

~~~query5~~~
Query optimized in 15 ms.
Owlim evaluation strategy : optimized in 15 ms.
Duration: 31 Result#: 0

~~~query6~~~
Query optimized in 0 ms.
Owlim evaluation strategy : optimized in 0 ms.
Duration: 0 Result#: 0

~~~query7~~~
Query optimized in 32 ms.
Owlim evaluation strategy : optimized in 32 ms.
Duration: 32 Result#: 0

~~~query8~~~
Query optimized in 0 ms.
Owlim evaluation strategy : optimized in 0 ms.
Duration: 0 Result#: 0

~~~query9~~~
Query optimized in 0 ms.
Owlim evaluation strategy : optimized in 0 ms.
Duration: 0 Result#: 0

~~~query10~~~
Query optimized in 0 ms.
Owlim evaluation strategy : optimized in 0 ms.
Duration: 0 Result#: 0

~~~query11~~~
Query optimized in 15 ms.
Owlim evaluation strategy : optimized in 15 ms.
Duration: 47 Result#: 0

~~~query12~~~
Query optimized in 0 ms.
Owlim evaluation strategy : optimized in 0 ms.
Duration: 0 Result#: 0

~~~query13~~~
Query optimized in 0 ms.
Owlim evaluation strategy : optimized in 0 ms.
Duration: 0 Result#: 0

~~~query14~~~
Query optimized in 0 ms.
Owlim evaluation strategy : optimized in 0 ms.
Duration: 39875 Result#: 3961133
### Finished testing OWLIM_500 ###

OWLIM_500
Time/Result#
query1 16 4
query2 84093 1284
query3 32 6
query4 15 0
```

```
query5 31 0
query6 0 0
query7 32 0
query8 0 0
query9 0 0
query10 0 0
query11 47 0
query12 0 0
query13 0 0
query14 39875 3961133
NumberOfStatements = 66731653
NumberOfExplicitStatements = 66731653
NumberOfEntities = 16429587
```

RDFS including Optimizations

```
Finished loading OWLIM_50 203 seconds

OWLIM_50 203 seconds

### Started testing OWLIM_50 ###

~~~query1~~~
Query optimized in 15 ms.
Owlim evaluation strategy : optimized in 15 ms.
Duration: 15 Result#: 4

~~~query2~~~
Query optimized in 47 ms.
Owlim evaluation strategy : optimized in 47 ms.
Duration: 1641 Result#: 130

~~~query3~~~
Query optimized in 16 ms.
Owlim evaluation strategy : optimized in 16 ms.
Duration: 31 Result#: 6

~~~query4~~~
Query optimized in 31 ms.
Owlim evaluation strategy : optimized in 31 ms.
Duration: 31 Result#: 34

~~~query5~~~
Query optimized in 0 ms.
Owlim evaluation strategy : optimized in 0 ms.
Duration: 16 Result#: 719

~~~query6~~~
Query optimized in 16 ms.
Owlim evaluation strategy : optimized in 16 ms.
Duration: 859 Result#: 393730

~~~query7~~~
Query optimized in 32 ms.
Owlim evaluation strategy : optimized in 32 ms.
Duration: 32 Result#: 59

~~~query8~~~
Query optimized in 47 ms.
Owlim evaluation strategy : optimized in 47 ms.
Duration: 172 Result#: 5916

~~~query9~~~
Query optimized in 46 ms.
Owlim evaluation strategy : optimized in 46 ms.
Duration: 2437 Result#: 6538

~~~query10~~~
Query optimized in 0 ms.
Owlim evaluation strategy : optimized in 0 ms.
Duration: 0 Result#: 0
```

```
~~~query11~~~
Query optimized in 0 ms.
Owlim evaluation strategy : optimized in 0 ms.
Duration: 0 Result#: 0

~~~query12~~~
Query optimized in 0 ms.
Owlim evaluation strategy : optimized in 0 ms.
Duration: 0 Result#: 0

~~~query13~~~
Query optimized in 0 ms.
Owlim evaluation strategy : optimized in 0 ms.
Duration: 0 Result#: 0

~~~query14~~~
Query optimized in 0 ms.
Owlim evaluation strategy : optimized in 0 ms.
Duration: 734 Result#: 393730
### Finished testing OWLIM_50 ###

OWLIM_50
Time/Result#
query1 15 4
query2 1641 130
query3 31 6
query4 31 34
query5 16 719
query6 859 393730
query7 32 59
query8 172 5916
query9 2437 6538
query10 0 0
query11 0 0
query12 0 0
query13 0 0
query14 734 393730
NumberOfStatements = 8318054
NumberOfExplicitStatements = 6655046
NumberOfEntities = 1639973
```

RDFS

```
Finished loading OWLIM_50 244 seconds

OWLIM_50 244 seconds

### Started testing OWLIM_50 ###

~~~query1~~~
Query optimized in 15 ms.
Owlim evaluation strategy : optimized in 15 ms.
Duration: 15 Result#: 4

~~~query2~~~
Query optimized in 47 ms.
Owlim evaluation strategy : optimized in 47 ms.
Duration: 2047 Result#: 130

~~~query3~~~
Query optimized in 31 ms.
Owlim evaluation strategy : optimized in 31 ms.
Duration: 47 Result#: 6

~~~query4~~~
Query optimized in 16 ms.
Owlim evaluation strategy : optimized in 16 ms.
Duration: 47 Result#: 34

~~~query5~~~
Query optimized in 16 ms.
Owlim evaluation strategy : optimized in 16 ms.
```

```
Duration: 16 Result#: 719

~~~query6~~~
Query optimized in 0 ms.
Owlim evaluation strategy : optimized in 0 ms.
Duration: 875 Result#: 393730

~~~query7~~~
Query optimized in 46 ms.
Owlim evaluation strategy : optimized in 46 ms.
Duration: 46 Result#: 59

~~~query8~~~
Query optimized in 32 ms.
Owlim evaluation strategy : optimized in 32 ms.
Duration: 235 Result#: 5916

~~~query9~~~
Query optimized in 31 ms.
Owlim evaluation strategy : optimized in 31 ms.
Duration: 3140 Result#: 6538

~~~query10~~~
Query optimized in 32 ms.
Owlim evaluation strategy : optimized in 32 ms.
Duration: 32 Result#: 0

~~~query11~~~
Query optimized in 0 ms.
Owlim evaluation strategy : optimized in 0 ms.
Duration: 15 Result#: 0

~~~query12~~~
Query optimized in 0 ms.
Owlim evaluation strategy : optimized in 0 ms.
Duration: 0 Result#: 0

~~~query13~~~
Query optimized in 0 ms.
Owlim evaluation strategy : optimized in 0 ms.
Duration: 0 Result#: 0

~~~query14~~~
Query optimized in 0 ms.
Owlim evaluation strategy : optimized in 0 ms.
Duration: 1063 Result#: 393730
### Finished testing OWLIM_50 ###

OWLIM_50
Time/Result#
query1 15 4
query2 2047 130
query3 47 6
query4 47 34
query5 16 719
query6 875 393730
query7 46 59
query8 235 5916
query9 3140 6538
query10 32 0
query11 15 0
query12 0 0
query13 0 0
query14 1063 393730
NumberOfStatements = 9401114
NumberOfExplicitStatements = 6655065
NumberOfEntities = 1639973
```

OWL-Horst including Optimizations

```
Finished loading OWLIM_50 433 seconds
```

```
OWLIM_50 433 seconds

### Started testing OWLIM_50 ###

~~~query1~~~
Query optimized in 16 ms.
Owlim evaluation strategy : optimized in 16 ms.
Duration: 16 Result#: 4

~~~query2~~~
Query optimized in 62 ms.
Owlim evaluation strategy : optimized in 62 ms.
Duration: 2094 Result#: 130

~~~query3~~~
Query optimized in 15 ms.
Owlim evaluation strategy : optimized in 15 ms.
Duration: 31 Result#: 6

~~~query4~~~
Query optimized in 31 ms.
Owlim evaluation strategy : optimized in 31 ms.
Duration: 47 Result#: 34

~~~query5~~~
Query optimized in 15 ms.
Owlim evaluation strategy : optimized in 15 ms.
Duration: 15 Result#: 719

~~~query6~~~
Query optimized in 0 ms.
Owlim evaluation strategy : optimized in 0 ms.
Duration: 1735 Result#: 519842

~~~query7~~~
Query optimized in 47 ms.
Owlim evaluation strategy : optimized in 47 ms.
Duration: 47 Result#: 67

~~~query8~~~
Query optimized in 31 ms.
Owlim evaluation strategy : optimized in 31 ms.
Duration: 218 Result#: 7790

~~~query9~~~
Query optimized in 47 ms.
Owlim evaluation strategy : optimized in 47 ms.
Duration: 4641 Result#: 13639

~~~query10~~~
Query optimized in 0 ms.
Owlim evaluation strategy : optimized in 31 ms.
Duration: 31 Result#: 4

~~~query11~~~
Query optimized in 0 ms.
Owlim evaluation strategy : optimized in 0 ms.
Duration: 0 Result#: 224

~~~query12~~~
Query optimized in 16 ms.
Owlim evaluation strategy : optimized in 16 ms.
Duration: 63 Result#: 15

~~~query13~~~
Query optimized in 0 ms.
Owlim evaluation strategy : optimized in 0 ms.
Duration: 469 Result#: 228

~~~query14~~~
Query optimized in 15 ms.
Owlim evaluation strategy : optimized in 15 ms.
Duration: 2984 Result#: 393730
```

```
### Finished testing OWLIM_50 ###

OWLIM_50
Time/Result#
query1 16 4
query2 2094 130
query3 31 6
query4 47 34
query5 15 719
query6 1735 519842
query7 47 67
query8 218 7790
query9 4641 13639
query10 31 4
query11 0 224
query12 63 15
query13 469 228
query14 2984 393730
NumberOfStatements = 11004153
NumberOfExplicitStatements = 6655055
NumberOfEntities = 1639978
```

OWL-Horst

```
Finished loading OWLIM_50 506 seconds

OWLIM_50 506 seconds

### Started testing OWLIM_50 ###

~~~query1~~~
Query optimized in 16 ms.
Owlim evaluation strategy : optimized in 16 ms.
Duration: 31 Result#: 4

~~~query2~~~
Query optimized in 63 ms.
Owlim evaluation strategy : optimized in 63 ms.
Duration: 2344 Result#: 130

~~~query3~~~
Query optimized in 15 ms.
Owlim evaluation strategy : optimized in 15 ms.
Duration: 15 Result#: 6

~~~query4~~~
Query optimized in 32 ms.
Owlim evaluation strategy : optimized in 32 ms.
Duration: 63 Result#: 34

~~~query5~~~
Query optimized in 16 ms.
Owlim evaluation strategy : optimized in 16 ms.
Duration: 16 Result#: 719

~~~query6~~~
Query optimized in 0 ms.
Owlim evaluation strategy : optimized in 0 ms.
Duration: 1515 Result#: 519842

~~~query7~~~
Query optimized in 47 ms.
Owlim evaluation strategy : optimized in 47 ms.
Duration: 47 Result#: 67

~~~query8~~~
Query optimized in 31 ms.
Owlim evaluation strategy : optimized in 31 ms.
Duration: 250 Result#: 7790

~~~query9~~~
Query optimized in 31 ms.
```

```
Owlim evaluation strategy : optimized in 31 ms.
Duration: 5531 Result#: 13639

~~~query10~~~
Query optimized in 0 ms.
Owlim evaluation strategy : optimized in 0 ms.
Duration: 0 Result#: 4

~~~query11~~~
Query optimized in 0 ms.
Owlim evaluation strategy : optimized in 0 ms.
Duration: 0 Result#: 224

~~~query12~~~
Query optimized in 32 ms.
Owlim evaluation strategy : optimized in 32 ms.
Duration: 94 Result#: 15

~~~query13~~~
Query optimized in 0 ms.
Owlim evaluation strategy : optimized in 0 ms.
Duration: 766 Result#: 228

~~~query14~~~
Query optimized in 0 ms.
Owlim evaluation strategy : optimized in 0 ms.
Duration: 3265 Result#: 393730
### Finished testing OWLIM_50 ###

OWLIM_50
Time/Result#
query1 31 4
query2 2344 130
query3 15 6
query4 63 34
query5 16 719
query6 1515 519842
query7 47 67
query8 250 7790
query9 5531 13639
query10 0 4
query11 0 224
query12 94 15
query13 766 228
query14 3265 393730
NumberOfStatements = 12087215
NumberOfExplicitStatements = 6655074
NumberOfEntities = 1639978
```

L2

```
Finished loading OWLIM_50 472 seconds

OWLIM_50 472 seconds

### Started testing OWLIM_50 ###

~~~query1~~~
Query optimized in 15 ms.
Owlim evaluation strategy : optimized in 15 ms.
Duration: 15 Result#: 4

~~~query2~~~
Query optimized in 125 ms.
Owlim evaluation strategy : optimized in 125 ms.
Duration: 2563 Result#: 130

~~~query3~~~
Query optimized in 15 ms.
Owlim evaluation strategy : optimized in 15 ms.
Duration: 31 Result#: 6
```

```
~~~query4~~~
Query optimized in 31 ms.
Owlim evaluation strategy : optimized in 31 ms.
Duration: 47 Result#: 34

~~~query5~~~
Query optimized in 0 ms.
Owlim evaluation strategy : optimized in 0 ms.
Duration: 15 Result#: 719

~~~query6~~~
Query optimized in 0 ms.
Owlim evaluation strategy : optimized in 0 ms.
Duration: 1282 Result#: 393730

~~~query7~~~
Query optimized in 31 ms.
Owlim evaluation strategy : optimized in 31 ms.
Duration: 31 Result#: 59

~~~query8~~~
Query optimized in 31 ms.
Owlim evaluation strategy : optimized in 31 ms.
Duration: 250 Result#: 5916

~~~query9~~~
Query optimized in 16 ms.
Owlim evaluation strategy : optimized in 16 ms.
Duration: 5187 Result#: 6538

~~~query10~~~
Query optimized in 0 ms.
Owlim evaluation strategy : optimized in 0 ms.
Duration: 0 Result#: 0

~~~query11~~~
Query optimized in 0 ms.
Owlim evaluation strategy : optimized in 0 ms.
Duration: 0 Result#: 224

~~~query12~~~
Query optimized in 0 ms.
Owlim evaluation strategy : optimized in 0 ms.
Duration: 0 Result#: 0

~~~query13~~~
Query optimized in 0 ms.
Owlim evaluation strategy : optimized in 0 ms.
Duration: 609 Result#: 228

~~~query14~~~
Query optimized in 0 ms.
Owlim evaluation strategy : optimized in 0 ms.
Duration: 3000 Result#: 393730
### Finished testing OWLIM_50 ###

OWLIM_50
Time/Result#
query1 15 4
query2 2563 130
query3 31 6
query4 47 34
query5 15 719
query6 1282 393730
query7 31 59
query8 250 5916
query9 5187 6538
query10 0 0
query11 0 224
query12 0 0
query13 609 228
query14 3000 393730
NumberOfStatements = 12371580
```

```
NumberOfExplicitStatements = 6655017
NumberOfEntities = 1639962

Finished loading OWLIM_200 2208 seconds

OWLIM_200 2208 seconds

### Started testing OWLIM_200 ###

~~~query1~~~
Query optimized in 16 ms.
Owlim evaluation strategy : optimized in 16 ms.
Duration: 16 Result#: 4

~~~query2~~~
Query optimized in 47 ms.
Owlim evaluation strategy : optimized in 47 ms.
Duration: 44187 Result#: 499

~~~query3~~~
Query optimized in 32 ms.
Owlim evaluation strategy : optimized in 32 ms.
Duration: 32 Result#: 6

~~~query4~~~
Query optimized in 62 ms.
Owlim evaluation strategy : optimized in 62 ms.
Duration: 78 Result#: 34

~~~query5~~~
Query optimized in 31 ms.
Owlim evaluation strategy : optimized in 31 ms.
Duration: 47 Result#: 719

~~~query6~~~
Query optimized in 15 ms.
Owlim evaluation strategy : optimized in 15 ms.
Duration: 21703 Result#: 1584743

~~~query7~~~
Query optimized in 15 ms.
Owlim evaluation strategy : optimized in 15 ms.
Duration: 15 Result#: 59

~~~query8~~~
Query optimized in 32 ms.
Owlim evaluation strategy : optimized in 32 ms.
Duration: 282 Result#: 5916

~~~query9~~~
Query optimized in 31 ms.
Owlim evaluation strategy : optimized in 31 ms.
Duration: 63734 Result#: 26227

~~~query10~~~
Query optimized in 0 ms.
Owlim evaluation strategy : optimized in 0 ms.
Duration: 0 Result#: 0

~~~query11~~~
Query optimized in 16 ms.
Owlim evaluation strategy : optimized in 16 ms.
Duration: 31 Result#: 224

~~~query12~~~
Query optimized in 0 ms.
Owlim evaluation strategy : optimized in 0 ms.
Duration: 0 Result#: 0

~~~query13~~~
Query optimized in 0 ms.
Owlim evaluation strategy : optimized in 0 ms.
Duration: 6188 Result#: 916
```

```
~~~query14~~~
Query optimized in 15 ms.
Owlim evaluation strategy : optimized in 15 ms.
Duration: 11890 Result#: 1584743
### Finished testing OWLIM_200 ###

OWLIM_200
Time/Result#
query1 16 4
query2 44187 499
query3 32 6
query4 78 34
query5 47 719
query6 21703 1584743
query7 15 59
query8 282 5916
query9 63734 26227
query10 0 0
query11 31 224
query12 0 0
query13 6188 916
query14 11890 1584743
NumberOfStatements = 49616662
NumberOfExplicitStatements = 26697033
NumberOfEntities = 6575127

Finished loading OWLIM_500 7920 seconds

OWLIM_500 7920 seconds

### Started testing OWLIM_500 ###

~~~query1~~~
Query optimized in 31 ms.
Owlim evaluation strategy : optimized in 31 ms.
Duration: 31 Result#: 4

~~~query2~~~
Query optimized in 47 ms.
Owlim evaluation strategy : optimized in 47 ms.
Duration: 136781 Result#: 1284

~~~query3~~~
Query optimized in 16 ms.
Owlim evaluation strategy : optimized in 16 ms.
Duration: 32 Result#: 6

~~~query4~~~
Query optimized in 47 ms.
Owlim evaluation strategy : optimized in 47 ms.
Duration: 78 Result#: 34

~~~query5~~~
Query optimized in 0 ms.
Owlim evaluation strategy : optimized in 15 ms.
Duration: 15 Result#: 719

~~~query6~~~
Query optimized in 0 ms.
Owlim evaluation strategy : optimized in 0 ms.
Duration: 28594 Result#: 3961133

~~~query7~~~
Query optimized in 16 ms.
Owlim evaluation strategy : optimized in 16 ms.
Duration: 16 Result#: 59

~~~query8~~~
Query optimized in 16 ms.
Owlim evaluation strategy : optimized in 16 ms.
Duration: 219 Result#: 5916
```

```
~~~query9~~~
Query optimized in 47 ms.
Owlim evaluation strategy : optimized in 47 ms.
Duration: 144672 Result#: 65638

~~~query10~~~
Query optimized in 31 ms.
Owlim evaluation strategy : optimized in 31 ms.
Duration: 47 Result#: 0

~~~query11~~~
Query optimized in 16 ms.
Owlim evaluation strategy : optimized in 16 ms.
Duration: 31 Result#: 224

~~~query12~~~
Query optimized in 16 ms.
Owlim evaluation strategy : optimized in 16 ms.
Duration: 16 Result#: 0

~~~query13~~~
Query optimized in 31 ms.
Owlim evaluation strategy : optimized in 31 ms.
Duration: 14890 Result#: 2370

~~~query14~~~
Query optimized in 16 ms.
Owlim evaluation strategy : optimized in 16 ms.
Duration: 29516 Result#: 3961133
### Finished testing OWLIM_500 ###

OWLIM_500
Time/Result#
query1 31 4
query2 136781 1284
query3 32 6
query4 78 34
query5 15 719
query6 28594 3961133
query7 16 59
query8 219 5916
query9 144672 65638
query10 47 0
query11 31 224
query12 16 0
query13 14890 2370
query14 29516 3961133
NumberOfStatements = 124005498
NumberOfExplicitStatements = 66731655
NumberOfEntities = 16429587
```

OWL-Max including Optimizations

```
Finished loading OWLIM_50 523 seconds

OWLIM_50 523 seconds

### Started testing OWLIM_50 ###

~~~query1~~~
Query optimized in 15 ms.
Owlim evaluation strategy : optimized in 15 ms.
Duration: 15 Result#: 4

~~~query2~~~
Query optimized in 47 ms.
Owlim evaluation strategy : optimized in 47 ms.
Duration: 2360 Result#: 130

~~~query3~~~
Query optimized in 15 ms.
Owlim evaluation strategy : optimized in 15 ms.
```

```
Duration: 15 Result#: 6

~~~query4~~~
Query optimized in 32 ms.
Owlim evaluation strategy : optimized in 32 ms.
Duration: 32 Result#: 34

~~~query5~~~
Query optimized in 15 ms.
Owlim evaluation strategy : optimized in 15 ms.
Duration: 15 Result#: 719

~~~query6~~~
Query optimized in 0 ms.
Owlim evaluation strategy : optimized in 0 ms.
Duration: 1485 Result#: 519842

~~~query7~~~
Query optimized in 31 ms.
Owlim evaluation strategy : optimized in 31 ms.
Duration: 31 Result#: 67

~~~query8~~~
Query optimized in 31 ms.
Owlim evaluation strategy : optimized in 31 ms.
Duration: 219 Result#: 7790

~~~query9~~~
Query optimized in 31 ms.
Owlim evaluation strategy : optimized in 31 ms.
Duration: 5531 Result#: 13639

~~~query10~~~
Query optimized in 0 ms.
Owlim evaluation strategy : optimized in 0 ms.
Duration: 0 Result#: 4

~~~query11~~~
Query optimized in 0 ms.
Owlim evaluation strategy : optimized in 0 ms.
Duration: 0 Result#: 224

~~~query12~~~
Query optimized in 125 ms.
Owlim evaluation strategy : optimized in 125 ms.
Duration: 203 Result#: 15

~~~query13~~~
Query optimized in 0 ms.
Owlim evaluation strategy : optimized in 0 ms.
Duration: 781 Result#: 228

~~~query14~~~
Query optimized in 0 ms.
Owlim evaluation strategy : optimized in 0 ms.
Duration: 2578 Result#: 393730
### Finished testing OWLIM_50 ###

OWLIM_50
Time/Result#
query1 15 4
query2 2360 130
query3 15 6
query4 32 34
query5 15 719
query6 1485 519842
query7 31 67
query8 219 7790
query9 5531 13639
query10 0 4
query11 0 224
query12 203 15
query13 781 228
```

```
query14 2578 393730
NumberOfStatements = 11149715
NumberOfExplicitStatements = 6655055
NumberOfEntities = 1639979

Finished loading OWLIM_200 2331 seconds

OWLIM_200 2331 seconds

### Started testing OWLIM_200 ###

~~~query1~~~
Query optimized in 31 ms.
Owlim evaluation strategy : optimized in 31 ms.
Duration: 31 Result#: 4

~~~query2~~~
Query optimized in 63 ms.
Owlim evaluation strategy : optimized in 63 ms.
Duration: 41797 Result#: 499

~~~query3~~~
Query optimized in 16 ms.
Owlim evaluation strategy : optimized in 16 ms.
Duration: 16 Result#: 6

~~~query4~~~
Query optimized in 47 ms.
Owlim evaluation strategy : optimized in 47 ms.
Duration: 62 Result#: 34

~~~query5~~~
Query optimized in 32 ms.
Owlim evaluation strategy : optimized in 32 ms.
Duration: 32 Result#: 719

~~~query6~~~
Query optimized in 0 ms.
Owlim evaluation strategy : optimized in 0 ms.
Duration: 44875 Result#: 2088195

~~~query7~~~
Query optimized in 46 ms.
Owlim evaluation strategy : optimized in 46 ms.
Duration: 46 Result#: 67

~~~query8~~~
Query optimized in 16 ms.
Owlim evaluation strategy : optimized in 16 ms.
Duration: 219 Result#: 7790

~~~query9~~~
Query optimized in 31 ms.
Owlim evaluation strategy : optimized in 31 ms.
Duration: 60781 Result#: 54285

~~~query10~~~
Query optimized in 0 ms.
Owlim evaluation strategy : optimized in 0 ms.
Duration: 0 Result#: 4

~~~query11~~~
Query optimized in 0 ms.
Owlim evaluation strategy : optimized in 0 ms.
Duration: 16 Result#: 224

~~~query12~~~
Query optimized in 31 ms.
Owlim evaluation strategy : optimized in 31 ms.
Duration: 1516 Result#: 15

~~~query13~~~
Query optimized in 0 ms.
```

```
Owlim evaluation strategy : optimized in 0 ms.
Duration: 5953 Result#: 916

~~~query14~~~
Query optimized in 0 ms.
Owlim evaluation strategy : optimized in 0 ms.
Duration: 10719 Result#: 1584743
### Finished testing OWLIM_200 ###

OWLIM_200
Time/Result#
query1 31 4
query2 41797 499
query3 16 6
query4 62 34
query5 32 719
query6 44875 2088195
query7 46 67
query8 219 7790
query9 60781 54285
query10 0 4
query11 16 224
query12 1516 15
query13 5953 916
query14 10719 1584743
NumberOfStatements = 44725123
NumberOfExplicitStatements = 26697071
NumberOfEntities = 6575144


Finished loading OWLIM_500 8183 seconds

OWLIM_500 8183 seconds

### Started testing OWLIM_500 ###

~~~query1~~~
Query optimized in 16 ms.
Owlim evaluation strategy : optimized in 16 ms.
Duration: 16 Result#: 4

~~~query2~~~
Query optimized in 47 ms.
Owlim evaluation strategy : optimized in 47 ms.
Duration: 109406 Result#: 1284

~~~query3~~~
Query optimized in 32 ms.
Owlim evaluation strategy : optimized in 32 ms.
Duration: 32 Result#: 6

~~~query4~~~
Query optimized in 46 ms.
Owlim evaluation strategy : optimized in 46 ms.
Duration: 62 Result#: 34

~~~query5~~~
Query optimized in 16 ms.
Owlim evaluation strategy : optimized in 16 ms.
Duration: 16 Result#: 719

~~~query6~~~
Query optimized in 15 ms.
Owlim evaluation strategy : optimized in 15 ms.
Duration: 60562 Result#: 5220814

~~~query7~~~
Query optimized in 47 ms.
Owlim evaluation strategy : optimized in 47 ms.
Duration: 47 Result#: 67

~~~query8~~~
Query optimized in 16 ms.
```

```
Owlim evaluation strategy : optimized in 16 ms.
Duration: 281 Result#: 7790

~~~query9~~~
Query optimized in 32 ms.
Owlim evaluation strategy : optimized in 32 ms.
Duration: 130953 Result#: 136165

~~~query10~~~
Query optimized in 32 ms.
Owlim evaluation strategy : optimized in 32 ms.
Duration: 32 Result#: 4

~~~query11~~~
Query optimized in 31 ms.
Owlim evaluation strategy : optimized in 31 ms.
Duration: 62 Result#: 224

~~~query12~~~
Query optimized in 31 ms.
Owlim evaluation strategy : optimized in 31 ms.
Duration: 47 Result#: 15

~~~query13~~~
Query optimized in 63 ms.
Owlim evaluation strategy : optimized in 63 ms.
Duration: 14219 Result#: 2370

~~~query14~~~
Query optimized in 0 ms.
Owlim evaluation strategy : optimized in 0 ms.
Duration: 28312 Result#: 3961133
### Finished testing OWLIM_500 ###

OWLIM_500
Time/Result#
query1 16 4
query2 109406 1284
query3 32 6
query4 62 34
query5 16 719
query6 60562 5220814
query7 47 67
query8 281 7790
query9 130953 136165
query10 32 4
query11 62 224
query12 47 15
query13 14219 2370
query14 28312 3961133
NumberOfStatements = 111799180
NumberOfExplicitStatements = 66731693
NumberOfEntities = 16429604
```

OWL-Max

```
Finished loading OWLIM_50 622 seconds

OWLIM_50 622 seconds

### Started testing OWLIM_50 ###

~~~query1~~~
Query optimized in 31 ms.
Owlim evaluation strategy : optimized in 31 ms.
Duration: 63 Result#: 4

~~~query2~~~
Query optimized in 47 ms.
Owlim evaluation strategy : optimized in 47 ms.
Duration: 2265 Result#: 130
```

```
~~~query3~~~
Query optimized in 16 ms.
Owlim evaluation strategy : optimized in 32 ms.
Duration: 47 Result#: 6

~~~query4~~~
Query optimized in 31 ms.
Owlim evaluation strategy : optimized in 31 ms.
Duration: 47 Result#: 34

~~~query5~~~
Query optimized in 0 ms.
Owlim evaluation strategy : optimized in 0 ms.
Duration: 16 Result#: 719

~~~query6~~~
Query optimized in 0 ms.
Owlim evaluation strategy : optimized in 0 ms.
Duration: 1375 Result#: 519842

~~~query7~~~
Query optimized in 15 ms.
Owlim evaluation strategy : optimized in 15 ms.
Duration: 31 Result#: 67

~~~query8~~~
Query optimized in 31 ms.
Owlim evaluation strategy : optimized in 31 ms.
Duration: 219 Result#: 7790

~~~query9~~~
Query optimized in 47 ms.
Owlim evaluation strategy : optimized in 47 ms.
Duration: 5375 Result#: 13639

~~~query10~~~
Query optimized in 0 ms.
Owlim evaluation strategy : optimized in 0 ms.
Duration: 0 Result#: 4

~~~query11~~~
Query optimized in 109 ms.
Owlim evaluation strategy : optimized in 109 ms.
Duration: 109 Result#: 224

~~~query12~~~
Query optimized in 16 ms.
Owlim evaluation strategy : optimized in 16 ms.
Duration: 63 Result#: 15

~~~query13~~~
Query optimized in 0 ms.
Owlim evaluation strategy : optimized in 0 ms.
Duration: 953 Result#: 228

~~~query14~~~
Query optimized in 0 ms.
Owlim evaluation strategy : optimized in 0 ms.
Duration: 3953 Result#: 393730
### Finished testing OWLIM_50 ###

OWLIM_50
Time/Result#
query1 63 4
query2 2265 130
query3 47 6
query4 47 34
query5 16 719
query6 1375 519842
query7 31 67
query8 219 7790
query9 5375 13639
query10 0 4
```

```
query11 109 224
query12 63 15
query13 953 228
query14 3953 393730
NumberOfStatements = 12232774
NumberOfExplicitStatements = 6655074
NumberOfEntities = 1639979
```

B.2 Wordnet Output

This appendix list the relevant result output of the test runs conducted using Wordnet, again organized according to the level of inference used during the run.

No Inference

```
Loaded: in 48 sec.
===== Shutting down ==========

NumberOfStatements = 1943045
NumberOfExplicitStatements = 1943045
NumberOfEntities = 841810
```

RDFS including Optimizations

```
Loaded: in 98 sec.
===== Shutting down ==========

NumberOfStatements = 4571025
NumberOfExplicitStatements = 1943075
NumberOfEntities = 841821
```

RDFS

```
Loaded: in 120 sec.
===== Shutting down ==========

NumberOfStatements = 5036048
NumberOfExplicitStatements = 1943094
NumberOfEntities = 841821
```

OWL-Horst including Optimizations

```
Loaded: in 259 sec.
===== Shutting down ==========

NumberOfStatements = 6163339
NumberOfExplicitStatements = 1943084
NumberOfEntities = 841826
```

OWL-Horst

```
Loaded: in 298 sec.
===== Shutting down ==========

NumberOfStatements = 6628353
NumberOfExplicitStatements = 1943103
NumberOfEntities = 841826
```

L2

```
Loaded: in 304 sec.
===== Shutting down ==========

NumberOfStatements = 7558124
NumberOfExplicitStatements = 1943047
NumberOfEntities = 841810
```

OWL-Max including Optimizations

```
Loaded: in 566 sec.
===== Shutting down ==========

NumberOfStatements = 8176412
NumberOfExplicitStatements = 1943084
NumberOfEntities = 841838
```

OWL-Max

```
Loaded: in 637 sec.
===== Shutting down ==========

NumberOfStatements = 8641440
NumberOfExplicitStatements = 1943103
NumberOfEntities = 841838
```