LDSR: Materialized Reason-able View to the Web of Linked Data

Atanas Kiryakov, Damyan Ognyanoff, Ruslan Velkov, Zdravko Tashev, Ivan Peikov

Ontotext AD, 135 Tsarigradsko Chaussee, Sofia 1784, Bulgaria
{first.second}@ontotext.com

Abstract. LDSR is a collection of datasets from the Linked Open Data (LOD) W3C community project, which have been selected and refined for the purpose of presenting a useful perspective to some of the central LOD datasets and to present a good use-case for large-scale reasoning and data integration. The design objectives are as follows: (i) consistency with respect to the formal semantics, (ii) generality – no specific domain knowledge should be required to comprehend most of the semantics, and (iii) heterogeneity – data from multiple data sources should be included. The current version of LDSR consists of about 440 million explicit statements and includes DBPedia, Geonames, Wordnet, CIA Factbook, lingvoj, and UMBEL. LDSR includes the ontologies of the datasets and the following schemata, used by them: SKOS, FOAF, RSS, and Dublin Core.

Here we report on the materialization of the deductive closure of LDSR performed with the OWLIM semantic repository, which uses a propriety native RDF rule engine. Entailment was performed with respect to a rule-set defining a tractable OWL dialect similar to OWL 2 RL, inferring 1.15 billion statements that have been materialized and indexed along the explicit ones. Although OWLIM performs complete forward-chaining, it does not materialize all the results for performance reasons. Groups of URIs, defined to be equivalent through **`owl:sameAs`**, are represented in the indices by a single super-node. Upon retrieval request, the repository "expands" the results in accordance with the **`owl:sameAs`** semantics. Thus, while the total number of all indexed statements is 1.58 billion, the number of retrievable statements is 2.32 billion.

The initial analysis of the results shows that the vast majority of the inferred statements match the expectations dictated by common sense. Although no formal validation has been performed, analysis of the ontologies and schemata used makes us believe that the OWL dialect used is sufficiently expressive, i.e. that reasoning with respect to a more expressive dialect will not entail additional implicit statements. There is still plenty of room left for analysis of the results and reasoning experiments with respect to various tasks (e.g. inconsistency checking) and OWL dialects. LDSR is available for exploration and query evaluation at http://www.ontotext.com/ldsr/.

1 Introduction

"Linked data" is defined by Tim Berners-Lee, [2], as RDF graphs, published so that they can be navigated across servers by following the links in the graph in a manner similar to the way the HTML web is navigated. The publishers of linked data should comply with four simple design principles:

1. Using URIs as names for things;
2. Using HTTP URIs, so that people can look up those names;
3. Providing useful information when someone looks up a URI;
4. Including links to other URI, so people can discover more things.

In fact, most of the RDF, [9], datasets fulfil principles 1, 2, and 4 by design. The piece of novelty in the design principles above concerns the requirement for enabling Semantic Web browsers to load HTTP descriptions of RDF resources based on their URIs. To this end, data publishers should make sure that:

- the "physical" addresses of the published pieces of data are the same as the "logical" addresses, used as RDF identifiers (URIs);
- upon receiving an HTTP request, the server should return an RDF-molecule, i.e. the set of triples that describe the resource.

Although not related to semantics, the linked data concept turns into an enabling factor for the realization of the Semantic Web as a global web of structured data around the Linking Open Data initiative introduced in section 1.1. Reasoning with linked data faces various obstacles related to the very scale and nature of such data. In order to provide context for the experiment presented in this paper, we provide, in section 1.2, a brief overview of the state of the art in scalable reasoning. In the second section, we propose the so-called reason-able views as a practical approach for reasoning with linked data.

The major contribution of this paper is a reason-able view called LDSR, presented in section 3, which allows experimenting with large-scale reasoning with general knowledge. Further, in sections 4 and 5, we share our experience gathered from the process of materialization of the deductive closure of LDSR, performed with the OWLIM semantic repository. Finally, we provide analysis of the results (section 6) and discussion on future work (section 7).

The results reported here are based on work performed within EC research projects RASCALLI and LarKC, in which LDSR is designed and used as a test-bed for scalable reasoning, [11], and for modelling of incomplete context-aware reasoning based on spreading activation and priming, [17]. The latter experiments were extended to usage of priming for pre-selection of relevant fractions of datasets for web-scale reasoning, [18].

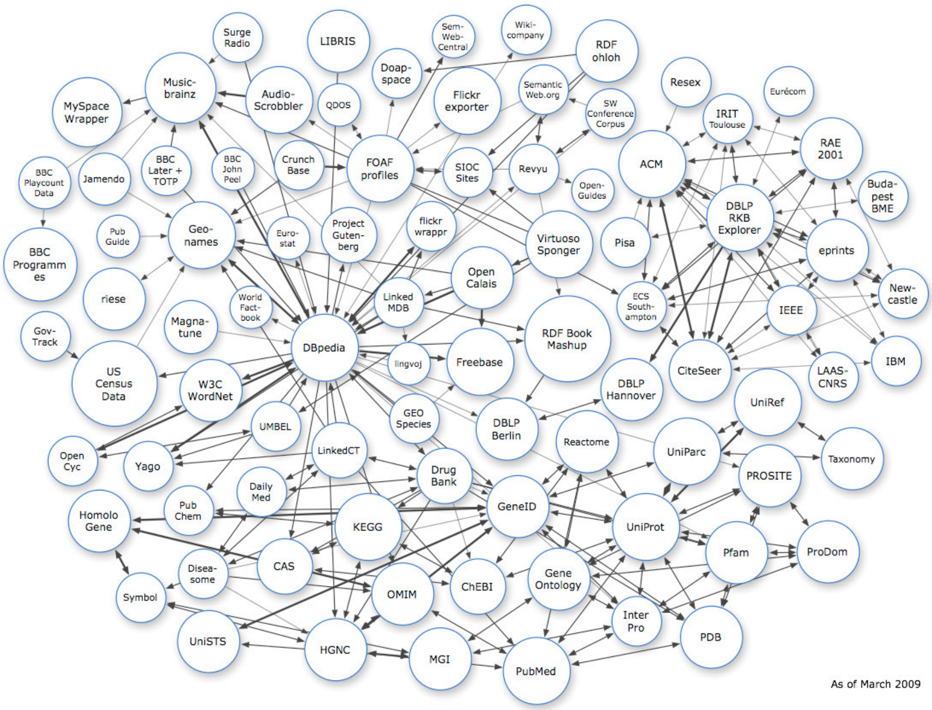
1.1 Linking Open Data

Linked Open Data (LOD[1]) is a W3C SWEO community project aiming to extend the Web by publishing open datasets as RDF and by creating RDF links between data items from different data sources. The central dataset of the LOD is DBPedia – an RDF extract of the Wikipedia open encyclopaedia. Because of the many mappings between other LOD datasets and DBPedia, the latter serves as a sort of a hub in the LOD graph assuring a certain level of connectivity. Among the LOD datasets are Wordnet, Geonames, World Factbook (see section 3 for information on these

[1] http://esw.w3.org/topic/SweoIG/TaskForces/CommunityProjects/LinkingOpenData

datasets), UniProt[2] (the largest integrated database with protein and gene-related information), FOAF[3] (a virtual collection of personal profiles), and OpenCyc[4] (the most popular upper-level and general ontology).

Currently LOD contains more than 40 datasets[5], with total volume above 4.7 billion statements, interlinked with 142 million statements as illustrated on Figure 1.



As of March 2009

Fig. 1. Map of the Datasets in Linking Open Data (LOD) Project
(from http://esw.w3.org/topic/SweoIG/TaskForces/CommunityProjects/LinkingOpenData)

1.2 Scalable Reasoning

For most of the popular knowledge representation (KR) formalisms and ontology languages, the worst case complexity of the algorithms for the basic reasoning tasks indicates that they are unfeasible for application to large scale knowledge bases and datasets. Still, as a result of the constant efforts on optimization of the reasoning

[2] http://www.uniprot.org/
[3] http://www.foaf-project.org/
[4] http://www.opencyc.org/
[5] http://esw.w3.org/topic/TaskForces/CommunityProjects/LinkingOpenData/DataSets

engines, many of them demonstrate impressive scalability and performance for a wide range of application scenarios.

Being the official schema definition and ontology languages of the Semantic Web, RDFS, [7], and OWL, [4], are by far the most popular KR languages nowadays. They standardize the epistemology, the vocabulary, and the syntax of the ontologies and the data encoded with respect to them. Yet, the semantics of RDFS and the various dialects of OWL are still quite diverse. As the computational characteristics of the different logical dialects vary dramatically, we will provide references only to few of the most scalable results for three classes of languages which seem to be most popular today and are most often employed for large scale reasoning.

Description Logics (DL): these provide balance between expressiveness and complexity of the reasoning algorithms; their most important advantage with respect to other fragments of the first order predicate calculus is that they are decidable. The most popular contemporary representative of this class is OWL DL, [4]. Still, the scalability of reasoning of the DL reasoners is limited by the fact that they are not tractable. The most scalable results are summarized in section 4.1 of [11]. In essence, the most scalable experiments with sound and complete OWL DL reasoning are in the range of 5 million statements, under the UOBM benchmark as reported in [8]. Inconsistency checking with respect to OWL DL has been performed, under specific constraints, against 60 million statements, as presented in [14].

F-Logic: a language, representative for a group of logical programming formalisms, which could be considered successors and PROLOG and Datalog. One can expect systems implementing F-logic-like languages to be easier to scale in comparison to the DL reasoners because rule-based entailment is of generally lower complexity, compared to the satisfiability checking performed by the DL reasoners. [14] presents results of comprehensive benchmarking with the OpenRuleBench benchmark suite performed on several of the most-popular rule engines. The largest scale experiments have been in the range between 5 and 10 million statements.

OWL Horst: we refer as “OWL Horst” a dialect of OWL defined in [16] as an extension of the RDFS semantics, [7], towards supporting some, but not all, OWL primitives. Ter Horst defines a rule language called R-entailment: both the body and the head of the rule are RDF graph patterns, described via statements, which can contain URIs, blank nodes, and variables in any position, as well as literals in the object position; blank nodes are not allowed in the body; all variables in the head of the rule should also appear in its body. The OWL dialect is defined as a set of R-entailment rules, named pD-entailment. OWL Horst is representative for a class of OWL dialects defined through R-entailment-like rule languages; most recently, OWL 2 RL was defined in [10] as a profile of OWL 2, based on a rule formalism almost identical to R-entailment. As presented in [12], there are plenty of systems (AllegroGraph, BigDATA, BigOWLIM, DAML DB, ORACLE) which can perform reasoning with respect to OWL Horst-like languages over datasets with a size in the range around one billion explicit statements. The most popular measuring stick for the performance of the engines at this scale is the 8000-university version of the LUBM benchmark, [5], referred to as LUBM(8000). Most of these engines perform total materialization of the deductive closure of the dataset during loading; loading, including materialization, can be performed on a commodity database server at a speed in the range of 20 000 explicit statements/second. If materialization is performed with respect to the RDFS semantics, LUBM(8000) is loaded twice faster; without any inference, speeds go up to 70 000 st./sec. In a scale-up experiment, BigOWLIM managed to load the 12 billion statements of LUBM(67000) and perform materialization at 12 000 st./sec.

To summarize, given all public results, only OWL Horst-like languages seem to be suitable for reasoning with data in range of billions of statements. Some engines (e.g. ORACLE, [19]) apply hybrid strategies in which DL reasoners are used to perform T-Box with the ontologies used in the datasets; the results are then materialized and used as input for OWL-Horst-style entailment.

2 Reasoning with Linked Data

Reasoning with linked data runs into various problems related to the clash of the mainstream reasoning techniques and the WWW-like nature of the data. The major issues can be summarized as follows:

- Most of the traditional reasoning setups implement sound and complete inference under the so-called “closed-world assumption”: the knowledge is considered complete; if specific fact is not known or inferable, it is not true. Such setups are irrelevant in environment where the knowledge is incomplete by design and logical consistency is not guaranteed;
- The complexity of reasoning with respect even to the simplest standard ontology languages (e.g. OWL Lite, [4]) is prohibitively high for the datasets in LOD (see section 1.2);
- Some of the datasets of LOD, or at least some parts of them, are not suitable for reasoning. It seems that many data publishers use OWL and RDFS vocabulary without accounting for its formal semantics;
- Some of the datasets are derived by the means of text-mining and, due to the intrinsic limitations of the accuracy of the extraction techniques, include incorrect information. For instance, the YAGO module of DBPedia contains plenty of faulty classifications of Wikipedia articles. Such inaccuracies are of relatively small number and probably not a serious problem for human readers exploring DBPedia. However, they can lead to significant noise and inconsistencies after reasoning;
- Although reasoning with data distributed across different WWW servers is possible, it is usually much slower than reasoning with local data.

Reason-able views represent an approach for reasoning with the web of linked data, introduced in [17]. We call *reason-able view (RAV)* an assembly of independent datasets, which can be used as a single body of knowledge (referred to as *integrated dataset*) with respect to reasoning and query evaluation. The integrated dataset represents the union of the independent datasets or versions of those, where parts of the original datasets could be excluded or refined in order to meet reasonability or some other criterion.

The notion of “reasonability” above means that the integrated dataset has certain specific qualities with respect to a specific reasoning task and language (more generally, specific deductive system). Examples for reasonability criteria could be “consistent with respect to OWL Lite” or “to allow RDFS entailment within O(n) time and space”. While in some scenarios one can find it useful to create a reasoning setup where different modules of the view can be subject to different reasoning, the simplest and easiest to use and manage is a setup where single criterion is used.

We define *linked data reason-able view* (*linked RAV*) as a reason-able view where:

- All the datasets in the view represent linked data (see section 1);
- Single reasonability criteria is imposed on all datasets;
- Each dataset is connected to at least one of the others.

Considering the size of the LOD datasets (see section 1.1), in order to make query evaluation and reasoning practically feasible, the integrated dataset of a linked RAV should be loaded in a single repository (even if it employs some sort of distribution internally). Such linked RAV can be considered as index, which caches parts of the LOD cloud and provides access to the datasets included in a manner similar to the one in which web search engines index WWW pages and facilitate their usage.

As a final practical consideration, to allow for caching and indexing, linked RAVs should include only datasets that are more or less static; this excludes various types of wrappers or virtual datasets, where RDF is generated in answer to retrieval requests (one can make an analogy with the dynamic part of the WWW).

3 Linked Data Semantic Repository (LDSR)

We defined LDSR as a reason-able view to the web of linked data, an assembly of some of the central LOD datasets, which have been selected and refined in order to:

- Serve as a useful index and entry point to the LOD cloud and
- Present a good use-case for large-scale reasoning and data integration.

The design objectives for LDSR were as follows:

1. Consistency with respect to the formal semantics;
2. Generality – no specific domain knowledge should be required to comprehend most of the semantics;
3. Heterogeneity – data from multiple data sources should be included;
4. Reasonability with respect to OWL 2 RL (see section 4 for details).

LDSR includes the following LOD datasets:

- **DBPedia**[6] is an RDF dataset derived from Wikipedia, designed and developed to provide as full as possible coverage of the factual knowledge that can be extracted from Wikipedia with a high level of precision. It serves as a hub for the LOD project.
- **Geonames**[7] is a geographic database that covers 6 million of the most significant geographical features on Earth (e.g. countries, populated places, mountains, rivers, and bridges), characterised by coordinates and relations to other features (e.g. “parent” feature in which the feature is nested).

[6] http://dbpedia.org/

[7] http://www.geonames.org/

- **UMBEL**[8] is a lightweight ontology structure, essentially, a hierarchy of about 20,000 classes, derived from OpenCyc and mapped to DBPedia. The classes range from general philosophical notions like **`TangibleThing`** to very specific classes like **`AbaCloth.`**
- **Wordnet**[9] is a lexical knowledge base that covers about 150,000 English words. Wordnet defines the meanings of English words by grouping them into sets of synonyms, called synsets. Each synset expresses a distinct concept. The words linked to a given synset are synonyms with respect to the meaning of the lexical concept represented by this synset. A word can have multiple meanings, i.e. it can be associated with multiple synsets. The more general terms are associated with less general terms through hyponym-hypernym relations. We use W3C's Wordnet RDF/OWL representation[10].
- **CIA World Factbook**[11] represents a collection of structured data, including statistical, geographic, political, and other information about all countries;
- **Lingvoj**[12] provides descriptions of the most popular human languages; currently it contains information about more than 500 languages.

The connectivity in LDSR is assured by DBPedia (which provides links to GeoNames, lingvoj, and Wordnet) and by UMBEL (which is linked to DBPedia). In LDSR, we include also the following ontologies and schemata, referred to or imported from the LOD datasets listed above:

- **Dublin Core**[13] (DC) is a relatively small but very popular metadata schema. It defines 15 attributes (e.g. author/contributor, date of publication, language, etc.) that can be used to describe information resources;
- **SKOS**[14] (Simple Knowledge Organization System) represents a relatively simple RDF schema that allows describing taxonomies of concepts linked to each other by any sort of subsumption hierarchy. The most important properties defined by SKOS are **`skos:broader`** and **`skos:narower`**, defined as inverse of each other. The subsumption semantics of these relationships is more appropriate for the encoding of "topic ontologies" and subjects classifiers as compared to **`rdfs:subClassOf`**.
- **RSS**[15] is an RDF schema designed to enable syndication of machine-readable information about updates from web sites;
- **FOAF**[16] is a project aimed at creating a network of machine-readable personal profiles published on the Web. In essence, the FOAF ontology defines the attributes of these personal profiles, which, in turn, allows for publication of contact information and links to other profiles.

[8] http://www.umbel.org/

[9] http://wordnet.princeton.edu/

[10] http://www.w3.org/2006/03/wn/wn20/

[11] http://www4.wiwiss.fu-berlin.de/factbook/

[12] http://www4.wiwiss.fu-berlin.de/factbook/

[13] http://purl.org/dc

[14] www.w3.org/2004/02/skos/

[15] http://web.resource.org/rss/1.0/spec

[16] http://www.foaf-project.org/

4 Reasoning Setup

The "reasonability criteria" (see section 2) for LDSR were defined with respect to OWL 2 RL. Formally, we wanted LDSR to allow forward-chaining, which means entailment and consistency checking, within O(n.log(n)) space and time. We wanted LDSR's integrated dataset to be consistent with respect to OWL 2 RL.

We also had one informal but very important objective towards the reasonability of LDSR: we wanted most of the results of the inference to comply with common sense without specific assumptions about the context of interpretation. In other words, we wanted to have a deductive closure that does not include statements which go against common sense, under the style and level of consensus similar to those of Wikipedia.

We used the BigOWLIM[17] semantic repository to load the datasets of LDSR and perform forward-chaining and materialization. This repository uses internally a rule language that supports R-entailment (see section 1.2) and can be configured to perform forward-chaining about predetermined rule-sets. The rule-set used for loading LDSR is the most expressive predefined rule-set of OWLIM, called "owl-max"; it extends OWL Horst, [16], to deliver expressiveness very similar to OWL 2 RL, [10].

The standard reasoning behaviour of OWLIM is to update the deductive closure upon committing of a transaction to the repository. Upon addition of statements, the new explicit statements are added to the repository in addition to the existing explicit statements that have come from the previous transactions and their closure. Forward-chaining with respect to the rules from the selected rule-set is performed in order to infer and add to the repository all statements that are inferable from the repository in its current state. This allows for efficient incremental updates of the deductive closure; one should consider that such procedure can deliver consistent results only for a monotonic reasoning system, such as R-entailment. Consistency checking is performed, applying the checking rules after adding all new statements and updating the deductive closure; in case of inconsistency, this is reported accordingly. Upon deletion of statements, the deductive closure is updated in order to withdraw statements that cannot be inferred from the new state of the repository.

4.1 Performance Tweaking of the RDFS and OWL Semantics

We load LDSR with OWLIM's "partialRDFS" option enabled, which excludes rules supporting some of the features from the semantic of RDFS and OWL, namely:

- `<X,rdf:type,rdf:Resource>` and `<P,rdf:type,rdf:Property>` statements are not being inferred respectively for all subject and predicates of statements;
- `<X,rdf:type,owl:Thing>` and `<X,owl:sameAs,X>` statements are not being inferred for all subjects of statements;
- `owl:Thing` and `owl:Nothing` are not asserted to be respectively super- and sub-classes of all classes.

[17] http://www.ontotext.com/owlim/

The above tweaks allow us to avoid inferring and indexing three “trivial” statements (as those above) for each URI in the repository. These modifications to the standard RDFS and OWL semantics are included also in LarKC’s minimal representation language, OWL-Lepton-I which is formally defined in section 4.3.1 of [6]. OWLIM’s “owl-max” can be regarded as an extension of OWL-Lepton-I towards OWL 2 RL.

4.2 owl:sameAs Optimizations

The loading of LDSR benefited greatly from a specific feature of the BigTRREE engine that allows the engine to handle efficiently **`owl:sameAs`** statements. **`owl:sameAs`** is a system predicate in OWL, declaring that two different URIs denote one and the same resource. Most often, it is used to align the different identifiers of the same real-world entity used in different data sources. For instance, in DBPedia, the URI of Vienna is http://dbpedia.org/page/Vienna, while in Geonames it is http://sws.geonames.org/2761369/. DBpedia contains the statement

```
(S1)  dbpedia:Vienna owl:sameAs geonames:2761369
```

which declares that the two URIs are equivalent. **`owl:sameAs`** is probably the most important OWL predicate when it comes to merging data from different data sources.

Following the formal definition of OWL (OWL 2 RL, to be more specific), whenever two URIs are declared equivalent, all statements that involve one of the URIs should be “replicated” with the other URI at the same position. For instance, in Geonames, the city of Vienna is defined as part of http://www.geonames.org/2761367/ (the first-order administrative division in Austria with the same name), which, in turn, is part of Austria (http://www.geonames.org/2782113):

```
(S2)  geonames:2761369 gno:parentFeature geonames:2761367
(S3)  geonames:2761367 gno:parentFeature geonames:2782113
```

As long as `gno:parentFeature` is a transitive relationship, in the course of the initial inference it will be derived that the city of Vienna is also part of Austria:

```
(S4) geonames:2761369 gno:parentFeature geonames:2782113
```

Due to the semantics of `owl:sameAs`, from (S1) it should be inferred that statements (S2) and (S4) also hold for Vienna when it is referred with its DBpedia URI:

```
(S5)  dbpedia:Vienna gno:parentFeature geonames:2761367
(S6)  dbpedia:Vienna gno:parentFeature geonames:2782113
```

These are true statements and when querying RDF data, no matter which one of the equivalent URIs is used in the explicit statements, the same results will be returned. When we consider that Austria, too, has an equivalent URI in DBpedia,

```
(S7) geonames:2782113 owl:sameAs dbpedia:Austria
```

we should also infer that:

```
(S8)  dbpedia:Vienna gno:parentFeature dbpedia:Austria
(S9)  geonames:2761369 gno:parentFeature dbpedia:Austria
(S10) geonames:2761367 gno:parentFeature dbpedia:Austria
```

In the above example, we had two alignment statements (S1 and S7), two statements carrying specific factual knowledge (S2 and S3), one statement inferred due to a transitive property (S4), and seven statements inferred as a result of **`owl:sameAs`** alignment (S5, S7, S8, S9, S10, and the inverse statements of S1 and S7). As we see, inference without **`owl:sameAs`** inflated the dataset by 25% (one new statement on top of 4 explicit), while **`owl:sameAs`** related inference increased the dataset by 175% (7 new statements). Considering that Vienna has an URI also in UMBEL, which is also declared equivalent to the one in DBpedia, the addition of one more explicit statement for this alignment, will cause inference of 4 new implicit statements (duplicates of S1, S5, S6, and S8). Although this is a small example, it provides a good indication about the performance implications of using **`owl:sameAs`** alignment in LOD. Also, because **`owl:sameAs`** is a transitive, reflexive, and symmetric relationship, a set of N equivalent URIs N^2 **`owl:sameAs`** statements will be generated for each pair of URIs (we should admit though that, in reality, there are not that many examples of large **`owl:sameAs`** equivalence classes). Thus, although **`owl:sameAs`** is useful for interlinking RDF datasets, its semantics causes considerable inflation of the number of implicit facts that should be considered during inference and query evaluation (either through forward- or through backward-chaining).

To overcome this problem, BigOWLIM handles **`owl:sameAs`** in a specific manner. In its indices, each set of equivalent URIs (equivalence class with respect to **`owl:sameAs`**) is represented by a single super-node. This way, BigTRREE does not inflate the indices and, at the same time, retains the ability to enumerate all statements that should be inferred using the equivalence upon retrieval request (e.g. during inference or query evaluation). Special care is taken to ensure that this "trick" does not hinder the ability to distinguish explicit from implicit statements.

5 Loading and Materialization Statistics

The statistics from loading and materialization of the implicit facts are presented in Table 1. The first column lists the datasets (or parts of them) in the order in which they were loaded into the repository. The number of triples listed in the Explicit Indexed Triples column indicates the increased number of statements in BigTRREE indices after the dataset has been loaded. Note that some data providers claim that their datasets contain an amount of statements, slightly different from the one presented in the table.

We can summarize the results of the loading of LDSR as follows:

- Number of inserted statements (NIS): 440 million;
- Number of stored statements (NSS), including the implicit ones: 1,585 mil.;

- Number of retrievable statements (NRS): 2,318 million.

The larger number of retrievable statements is a result of the **owl:sameAs** optimization discussed in section 4.2; the optimization has “compressed” 734 million statements, reducing the size of the indices by 32%. Each explicit triple caused, on average, the materialization and indexing of 2.6 new implicit triples. There are 5.3 triples “retrievable” against a single explicit statement asserted.

Loading of LDSR, including forward-chaining, materialization and full-text indexing of the literals, took BigOWLIM 3.1 almost 34 hours; which suggests loading speed of 3600 explicit statements/second. The test is performed using a server with the following specifications: 2 x Xeon 5420 CPU (2.5 GHz), 64GB of RAM, OpenSolaris, JDK 1.6, RAID array of 8 SAS drives in RAID 5. In another run, we have managed to load LDSR on a desktop machine with 12 GB of RAM, but it took a longer, due to smaller cache capacities.

Table 1: LDSR loading and inference statistics.

Dataset	**Explicit Indexed Triples** ('000)	**Inferred Indexed Triples** ('000)	**All Indexed Triples** ('000)	**Entities** ('000 of nodes in the graph)	**Infer-red closure ratio**
Schemata and ontologies	10	7	17	5	0.7
DBPedia (SKOS categories)	2,233	262,734	264,968	952	117.6
DBpedia (owl:sameAs)	2,053	4,006	6,059	4,005	2.0
UMBEL	3,197	41,228	44,425	1,388	12.9
Lingvoj	20	112	132	18	5.7
CIA Factbook	161	40	202	53	0.2
Wordnet	1,943	5,236	7,179	842	2.7
Geonames	72,749	471,220	543,969	33,382	6.5
DBpedia core	357,450	360,172	717,621	85,998	1.0
Total	**439,815**	**1,144,755**	**1,584,571**	**126,642**	**2.6**

The current version of LDSR includes version 3.3 of DBPedia and a version of Geonames downloaded in March 2009.

6 Analysis of the Results

The most important outcome of this experiment is that it showed it was possible to build a reason-able view that matches the requirements set forth in section 3:

- LDSR really integrates into a single body of knowledge several of the central datasets in LOD. It contains common sense knowledge by design;

- LDSR contains quite heterogeneous datasets. The nature of the knowledge encoded in them varies from encyclopaedic (DBPedia), through geographic (Geonames), to linguistic (Wordnet and lingvoj) and taxonomical (UMBEL).
- The vast majority of the facts inferred from the knowledge in LDSR look reasonable and does not go against common sense and the knowledge we have from life experience; we draw this conclusion from our practice of intensively exploring and querying the LDSR over the last several months. The only exception are the SKOS categories in DBPedia, discussed below.
- The integrated dataset of LDSR is logically consistent.

The size of the deductive closure allows for its efficient indexing and maintenance together with the explicit knowledge in the same repository. The tweaking of the RDFS and OWL semantics (see section 4.1) allowed us to avoid the materialization of about 200 million "trivial" statements. Without such tweaking and without the **owl:sameAs** optimization (see section 4.2), the inferred closure would have been twice bigger, which is still a manageable size.

In most of the cases, the high ratio of expansion of the deductive closure were due to long chains of statements over transitive properties, that are used to construct hierarchies. This is the case with the nesting of locations over the **gno:parentFeature** in Geonames, the class hierarchy in UMBEL, and the category hierarchy in DBPedia.

6.1 Fixing the category hierarchy of DBPedia

The gravest problem we faced with respect to ensuring "reason-ability" for LDSR was related to the category hierarchy in DBpedia. This hierarchy includes 478 thousand categories linked with 897 thousand relations; the categories are used for classification of articles/entities in Wikipedia and, as a result, in DBpedia. While the hierarchy is defined via **skos:broader** relations, in many of the cases the actual relationship is, in a general context, either too weak and insignificant or simply inaccurate. Quite often concepts, the meanings of which were overlapping, were incorrectly encoded as a pair of boarder-narrower categories, instead of just related categories. Combined with the extensive usage of auxiliary categories and multiple-inheritance, this resulted in extremely tangled hierarchy which even contained cycles of categories related through transitive subsumption relationships. The result of such cycles is that after materialization all categories in the cycle become equivalent to one another. During this experiment 2,165 simple cycles were detected, 1,321 of which were trivial (a category being marked as broader to itself); the latter were instantly discarded. In order to "fix" the remaining 844 non-trivial cycles, a member of the team have analyzed all the them and changed 868 relations from **skos:broader** to **skos:related**. The resulting graph contained **skos:broader** paths of lengths ranging from 1 to 177.

In order to bring the category hierarch of DBPedia to a reason-able form, we performed several further refinements, as reported in [12]. The resulting dataset contains 728,882 **skos:broader** and 582,087 **skos:related** statements. The inferred closure of the **skos:broader** relations contains around 262 million SKOS-related statements with predicates `broader`, `narrower`, `related`, `broaderTransitive`, `narowerTransitive`, and `semanticRelation`.

6.2 Differences between LDSR and LUBM with respect to inference

Generally, one can observe that reasoning with real-world data appears to be much more challenging, compared to synthetic tests like LUBM, [5]. The differences between LDSR's integrated dataset and the datasets generated and used in LUBM can be summarized as follows:

- The RDF graph in LDSR has star-like topology: the sub-graph for each university is connected only to the sub-graphs of the LUBM ontology and the first university which stand in the centre of the "star". This allows for easy partition and caching in the process of loading LDSR. In contrast, the graph of LDSR is highly interconnected and there is no easy way to isolate and cache only the most used parts of.
- The deductive closure of LUBM expands the indices by 70%, while in LDSR the expansion is 260%. The major reason for this are the long chains of predicates related over transitive properties and the intensive usage of `owl:sameAs`.
- In LDSR there are more than 100,000 different predicates used, mostly due to the encoding style of DBPedia. On the other hand, in LUBM there are just handful of predicates used, which allows for efficient loading and querying of LUBM in repository configurations where indices with predicates as primary sorting criteria are not maintained.

7 Conclusion and Future Work

We managed to select and refine several of the central datasets from the LOD data cloud and to load them in the OWLIM semantic repository. Forward-chaining inference was performed to materialize the deductive closure; as a result, 1.1 billion implicit statements were materialized from 440 million explicit ones and indexed, bringing the total size of the repository to 1.5 billion triples. About 734 million statements, inferable on the basis of `owl:sameAs` equivalence, are not materialized; they are rather "generated" upon retrieval. Thus, the total number of the statements retrievable from the LDSR repository goes up to 2.3 billion.

The initial analysis of the results shows that the vast majority of the inferred statements match the common sense expectations. Although no extensive formal validation has been performed, our analysis of the ontologies and schemata used in the selected datasets makes us believe that the OWL dialect used is sufficiently expressive to unveil their complete semantics. In other words, we believe that reasoning with respect to a more expressive dialect will not entail additional implicit statements.

There is still plenty of room left for analysis of the results and experiments with respect to various reasoning tasks (e.g. inconsistency checking) and OWL dialects.

One immediate goal is to perform entailment with respect to the normative OWL 2 RL rules in order to confirm the reason-ability of LDSR with respect to it and further refine the datasets, if necessary. On the usability site, we are experimenting with few applications of LDSR, e.g. semantic annotation of text with respect to the entities in LDSR or using it for query expansion for services like Flicker.

To the best of our knowledge, LDSR is the largest body of general knowledge (not specific to a particular scientific domain) that someone has ever performed inference against. The only larger reason-able dataset that we know is the Pathway and Interaction Knowledge Base (PIKB, available at http://www.linkedlifedata.org). PIKB is the second reason-able view to the web of linked data developed by Ontotext. It assembles a large fraction of the life-science-related datasets in LOD, including about 20 databases, as documented in [1]. PIKB includes about 1.5 billion explicit triples, which are complemented by another 842 million implicit statements inferred from them.

We maintain a public demonstration service, available at http://www.ontotext.com/ldsr/, which allows one to explore LDSR and evaluate queries against it through a web interface. Programs can use LDSR through a SPARQL end-point.

References

1. Andersson, B., Momtchev, V.: *LarKC Requirements summary and data repository.* LarKC project deliverable D7a.1.1, 2008.
2. Berners-Lee, T. (2006). *Design Issues: Linked Data*. http://www.w3.org/DesignIssues/LinkedData.html
3. Brickley, D., Guha, R.V, eds. (2004). *Resource Description Framework (RDF) Schemas.* W3C Recommendation. http://www.w3.org/TR/2004/REC-rdf-schema-20040210/.
4. Dean, M; Schreiber, G. – editors; Bechhofer, S; van Harmelen, F; Hendler, J; Horrocks, I.; McGuinness, D. L; Patel-Schneider, P. F.; Stein, L. A. (2004). *OWL Web Ontology Language Reference*. W3C Recom., 10 Feb. 2004. http://www.w3.org/TR/owl-ref/.
5. Guo, Y; Pan, Z; and Heflin, J. (2004). *An Evaluation of Knowledge Base Systems* for *Large OWL Datasets.* Journal of Web Semantics, 3(2), 2005, pp. 158-182. http://www.websemanticsjournal.org/ps/pub/2005-16
6. Fischer, F; Keller, U; Kiryakov, A; Huang, Z; Momtchev, V; Simperl, E. (2008). *Initial Knowledge Representation Formalism.* LarKC project deliverable D1.1.3.
7. Hayes, P. (2004). *RDF Semantics*. W3C Recommendation 10 Feb. 2004. http://www.w3.org/TR/2004/REC-rdf-mt-20040210/
8. Ma, L; Yang, Y; Qiu, Z; Xie, G; Pan, Y. (2006) *Towards A Complete OWL Ontology Benchmark*. In Proc. of the 3rd European Semantic Web Conference (ESWC 2006). Budva (Montenegro).
9. Manola F., Miller, E. (eds.) (2004). *RDF Primer. W3C Recommendation 10 Feb 2004,* http://www.w3.org/TR/REC-rdf-syntax/
10. Motik, B; Cuenca Grau, B; Horrocks, I; Wu, Z; Fokoue, A; Lutz, C. (eds.) (2009). *OWL 2 Web Ontology Language Profiles*. W3C Candidate Recommendation 11 June 2009. http://www.w3.org/TR/owl2-profiles/
11. Kiryakov, A. (2008). *Measurable Targets for Scalable Reasoning.* LarKC project deliverable D5.5.1. http://www.larkc.eu/deliverables/

12. Kiryakov, A; Tashev, Z; Ognyanoff, D; Velkov, R; Momtchev, V; Balev, B; Peikov, I. (2009). *Validation goals and metrics for the LarKC platform*. LarKC project deliverable D5.5.2. http://www.larkc.eu/deliverables/
13. Liang, S.; Fodor, P.; Wan, H; Kifer, M. (2009). *OpenRuleBench: An Analysis of the Performance of Rule Engines*. In Proc. of the 18th International World Wide Web Conference (WWW 2009), Mardid.
14. Schonberg, E; Srinivas, K; Kalyanpur, A; Cimino, J; Patel, C; Dolby, J; Kershenbaum, A; Ma, L; Fokoue, A. (2007). *Matching Patient Records to Clinical Trials Using Ontologies*. In Proc. ISWC 2007.
15. Ontotext Lab. (2007). *SwiftOWLIM. System Documentation.* Version 2.9.1 from 30 Sep, 2007. http://www.ontotex.com/owlim/
16. ter Horst, H. J. (2005) *Combining RDF and Part of OWL with Rules: Semantics, Decidability, Complexity*. In Proc. of ISWC 2005, Galway, Ireland, November 6-10, 2005. LNCS 3729, pp. 668-684.
17. Todorova, P., Kiryakov, A., Ognyanoff, D., Peikov, I., Velkov, R., Tashev, Z. (2009). *Spreading Activation Components.* LarKC project deliverable D2.4.1. http://www.larkc.eu/deliverables/
18. Velkov, R., Ognyanoff, D., Kiryakov, A. (2009). *Open-Domain Incomplete Reasoner.* RASCALLI project deliverable *D3b.* http://www.ofai.at/rascalli/
19. Wu, A.; Lopez, X.: *Building Enterprise Applications With Oracle Database 11g Semantic Technologies.* Presentation at Semantic Technologies Conference, San Jose, 2009.