U2R2 – The Ulm University Relational Reasoner: System Description

Timo Weithöner

Inst. of AI, Ulm University, 89069 Ulm, Germany
timo.weithoener@uni-ulm.de

Abstract. This is a system description of the Ulm University Relational Reasoner (U2R2). The system merges rule based DL reasoning with technologies from relational database management systems. U2R2 implements a total forward chaining and materialization approach, which calculates and persistently stores all possible inferences whenever a knowledge base is loaded or altered. As a result U2R2 offers excellent query response times for TBox as well as ABox queries. The system is not limited by main memory restrictions as it leverages secondary storage, which allows to process huge knowledge bases even on standard desktop computers. Additional features include incremental reasoning, retraction, and availability of savepoints, which allow to restore previous system states.

1 Motivation

Today a growing number of applications such as reasoning over social networks or GUI-based ABox exploration [1] require excellent query response times even when faced with huge ontologies. This is especially true for the latter example where multiple ontology queries might be triggered by a single mouse click during browsing. Here the query response times often need to be below 10ms to enable users to work fluently. Also main memory restrictions are likely to interfere with the reasoning process when dealing with huge ontologies. Such a setting requires a reasoner which leverages secondary storage and which is optimized for fast query answering.

In addition interactive ontology applications will be characterized by small but frequent ontology changes. Incremental reasoning and delete operations (retraction) should thus be supported.

This paper describes U2R2 an ontology reasoner which uses database technologies to address the above challenges. Section 2 provides an insight into U2R2's architecture while Section 3 offers initial evaluation results. Section 4 highlights some related work and Section 5 concludes this paper.

2 The U2R2 System

The development of U2R2 was motivated by the application requirements described above. The system merges DL reasoning techniques with technologies known from the field of deductive and relational database management systems (DedDBMS and RDBMS). U2R2 is implemented in Java and uses the Web Ontology Language (OWL)for ontology uploads.

Reasoning Strategy. U2R2 implements a rule based reasoning approach with total forward chaining and materialization. This means that whenever axioms are added to the ontology each axiom is translated into a set of base facts (i.e. tuples in specific relations). Then the integrated rule engine applies rules on these facts until a fix point is reached and no more new facts can be inferred. During this process all generated facts are persistently stored for future reasoning and query answering.

The rule set used by U2R2 is based on the "meta mapping" described in [2]. Meta mapping rules are a slight improvement of the standard rule set suggested for DLP [3]. Within the standard approach every ontology axiom becomes a rule while class or property assertions result in base facts. This means that the rule set is different for every ontology. Also there is no straightforward way of answering class subsumption queries [2]. In contrast the rule set used in U2R2 (meta mapping) is identical for all ontologies. All ontology axioms and assertions are translated into base facts of the respective predicates (see Table 1 for an example). U2R2 currently support the $\mathcal{SHIF}$ language fragment.

Table 1. Conversion of a Range Restriction into DLP and Meta Mapping

Abstr. DL	DLP	Meta Mapping

$\top \sqsubseteq \forall P.C$	`C(F) :- R(I, F).`	`range("R", "C").`
		`type(F, C) :-`
		`propInst(P, I, F), range(P, C).`

To evaluate the rules they are converted into a relational algebra program, where each predicate symbol (e.g. `type`) is represented by a relation. Each fact can be understood as an entry in such a base relation. Rules correspond to insertions into the head relation. The set of tuples to be inserted is computed from the rule body. Here joins are created from conjunctions over predicates with equal variable names and instantiated terms are converted into selections. Recursion is evaluated using delta iteration. Finally, the transition from rule body to head becomes a projection.

U2R2's rule set is fully customizable. It can be tailored to the needs of specific applications. This means that predefined rules may be deleted or additional rules and relations can be added to the system.

U2R2's DBMS Heritage. All ontology data, namely axioms and assertions, loaded into U2R2 are stored in relations. Unlike a relational database all relations share the same fixed schema. U2R2 relations accommodates tuples with up to three attributes. A number of relational algebra operations have been implemented on these relations. These operations are: Selection, projection, union, join and set difference. In contrast to RDBMS U2R2 silently enforces uniqueness of tuples in relations. The system ignores duplicates without aborting transactions in case a unique constraint would be violated.

OWL identify concepts, properties and individuals using URIs. Consequently all attributes of all relations represent URIs. To speed up operations on relations and to keep relations compact, numeric identifiers for URIs are kept in the relations. A separate map which links these ids to the real URI (and vice versa) is maintained by the system.

All data contained in relations and the aforementioned map is stored in pages. Pages are equally sized chunks of memory which usually reside in secondary storage. Only if the data contained in a certain page is required for processing the page is loaded into main memory. Thereby U2R2 only keeps a limited number of pages in main memory. In case this limit is exceeded other pages are swapped to disk. As a result no matter how many tuples are stored in the system the total main memory consumption is limited.

As in commercial RDBMS the system allows to define and restore savepoints. This offers a very efficient way to roll back to a consistent ontology in case changes led to inconsistencies. Management of pages, logging, index structures and join optimization are also using techniques comparably found in RDBMS.

Query Processing. U2R2's reasoning strategy of total forward chaining and materialization means that all reasoning is done immediately after loading or altering an ontology. Thereby all inferred knowledge is stored persistently. As a consequence basic queries like class hierarchy queries or class instance retrieval can be mapped to simple selections. Since selects are supported by indexes these queries are processed extremely fast. A comprehensive set of such queries is offered via a Java API.

Conjunctive queries are also supported by U2R2. The availability of joins, selections and projections on relations containing the materialized reasoning results enables to code such queries using U2R2's Java API. However as of today U2R2 is not capable of interpreting a conjunctive query language such as SPARQL[1].

3 System Evaluation

The U2R2-System is fully implemented and was successfully connected to the OntoTrack ABox explorer [1] and provides reasoning services in an application which is exploring a movie ontology consisting of data about 20.204 movies and 149.397 related persons (actors, directors, ...).

Initital spot tests were conducted with the Lehigh University Benchmark (LUBM) [4]. We compared U2R2 with OWLIM and KAON2 and also included

[1] `http://www.w3.org/TR/rdf-sparql-query/`

two tableaux style systems namely RacerPro and Pellet. Tests were conducted on a standard Desktop machine (3 GHz, 1 GB RAM, Windows XP, Java VM limited to 750 MB of heap).

Unsurprisingly U2R2 is capable of loading 10 Universities with 15 Departments each (for comparison, Pellet only managed 7 universities). But U2R2 is by far the slowest system when loading an ontology (about 20 times slower than the fastest system, OWLIM). However U2R2 outperforms all other systems when processing queries. This can be seen when looking at LUBM Query 9, which is one of the most challenging queries in the benchmark (Pellet was unable to answer the query for more than 3 and RacerPro for more than 5 universities). Here U2R2 shows minimal query times and is even faster then OWLIM (see Figure 1).

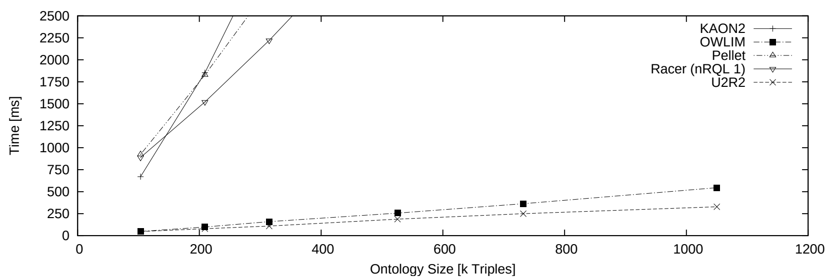

Fig. 1. Comparing query execution times for LUBM query 9

4 Related Work

Unlike Racer LAS [5] or the Instance Store [6] U2R2 is not a reasoner which is accompanied by a relational database. Instead U2R2 is a reasoning engine built inside a highly specialized relational database core. This also differentiates U2R2 from KAON2 [7], a disjunctive Datalog engine, which claims to be a DedDBMS but doesn't leverage secondary storage in the reasoning process (which is the same with "historic" deductive databases like LOLA [8] or CORAL [9]). The HAWK system (formerly DLDB)[2] implements an ontology repository which may utilize a RDMBS for storage but has to connect to a remote reasoner for inference services. Another related system is QuOnto [10]. Here a relational database is used for the processing of conjunctive queries. However QuOnto only supports ABox queries and does not materialize reasoning or query results. The Oracle Spatial Resource Descriptor Framework [11] is an RDF storage which also features rule based RDFS inference. RDFS is significantly less expressive than the language fragment supported by U2R2. The most similar system to U2R2 is OWLIM [12], which follows a comparable reasoning strategy but is limited to reasoning in main memory.

[2] http://swat.cse.lehigh.edu/downloads/hawk.html

5 Conclusion and Outlook

This paper presented the U2R2 reasoner. To the best of our knowledge U2R2 is the only available system which combines DL reasoning in the $\mathcal{SHIF}$ language fragment, incremental reasoning and materialization of reasoning results with database features such as persistence, leverage of secondary storage and the availability of savepoints.

We plan to further improve reasoning performance by parallelization of rule processing. Also additional interfaces (like support for SPARQL) are planned to increase usability of the system.

References

1. Noppens, O., Liebig, T.: Understanding large volumes of interconnected individuals by visual exploration. In Franconi, E., Kifer, M., May, W., eds.: ESWC. Volume 4519 of Lecture Notes in Computer Science., Springer (2007) 799–808
2. Weithöner, T., Liebig, T., Specht, G.: Storing and Querying Ontologies in Logic Databases. In: Proceedings of the Workshop "Semantic Web and Databases" at the VLDB 2003, Berlin, Germany (2003) Online Proceedings.
3. Grosof, B.N., Horrocks, I., Volz, R., Decker, S.: Description Logic Programms: Combining Logic Programms with Description Logic. In: Proceedings of the 12^{th} International World Wide Web Conference, Budapest, Hungary (2003)
4. Guo, Y., Pan, Z., Heflin, J.: LUBM: A benchmark for OWL knowledge base systems. Journal of Web Semantics **3** (2005) 158–182
5. Chen, C., Haarslev, V., Wang, J.: LAS: Extending Racer by a Large ABox Store. In Horrocks, I., Sattler, U., Wolter, F., eds.: Description Logics. Volume 147 of CEUR Workshop Proceedings., CEUR-WS.org (2005)
6. Bechhofer, S., Horrocks, I., Turi, D.: The OWL Instance Store: System Description. In Nieuwenhuis, R., ed.: CADE. Volume 3632 of LNCS., Springer (2005) 177–181
7. Motik, B., Studer, R.: KAON2 – A Scalable Reasoning Tool for the Semantic Web. In: Proceedings of the 2nd European Semantic Web Conference (ESWC'05), Heraklion, Greece (2005)
8. Freitag, B., Schütz, H., Specht, G.: *LOLA* - A Logic Language for Deductive Databases and its Implementation. In: Procc. 2^{nd} International Symposium on Database Systems for Advanced Applications (DASFAA '91), Tokyo, Japan (1991) 216–225
9. Ramakrishnan, R., Srivastava, D., Sudarshan, S., Seshadri, P.: The CORAL Deductive System. VLDB Journal: Very Large Data Bases **3** (1994) 161–210
10. Acciarri, A., Calvanese, D., Giacomo, G.D., Lembo, D., Lenzerini, M., Palmieri, M., Rosati, R.: QuOnto: Querying Ontologies. In Veloso, M.M., Kambhampati, S., eds.: AAAI, AAAI Press / The MIT Press (2005) 1670–1671
11. Murray, C.: Oracle Spatial, Resource Description Framework (RDF), 10g Release 2 (10.2). Technical report, Oracle (2005)
12. Kiryakov, A., Ognyanov, D., Manov, D.: OWLIM — A Pragmatic Semantic Repository for OWL. In: Proc. of the Int. Workshop on Scalable Semantic Web Knowledge Base Systems (SSWS'05). Volume 3807 of LNCS., New York City, USA, Springer (2005) 182–192