Universität Ulm

Fakultät für Informatik

Reasoning with OWL
- System Support and Insights -

Thorsten Liebig
Universität Ulm

Nr. 2006-04

Ulmer Informatik-Berichte

September 2006

Informatik Bericht Nr. 2006-04
(Technical report 2006-04, Computer Science Faculty, Ulm University)

Reasoning with OWL[1]
– System Support and Insights –

Thorsten Liebig
Ulm University

September 2006

This report aim at summarizing the current activities around OWL, the Web Ontology Language. At first, the report will present details about the current effort towards a revision of the official OWL W3C recommendation, known as OWL 1.1. Secondly, it describes a selection of inference engines while discussing different approaches as well as conceptual limits. These systems are then empirically evaluated using a set of spot tests which are intentionally designed to be hard to solve but small in size. Thirdly, it discusses actual trends and forthcoming developments in the context of ontology development and ontology reasoning. As a whole this report tries to provide some insights into currently available reasoning systems in order to serve as a decision help for Semantic Web application designers.

[1]This report is a result of a collaboration between the Department of Artificial Intelligence at the University of Ulm and DoCoMo Euro-Labs, Munich, Germany. Appeared also as DoCoMo Euro-Labs Internal Technical Report I-FN-83, October 2006

Contents

1. Introduction

OWL, the W3C Web Ontology Language, has now been a W3C recommendation since more than two years. It starts to play an increasingly important role in the business field of semantic technologies. For instance, according to a recent market report at least 200 business entities are currently engaged in **semantic technology** R&D on a market which is expected to grow 10-fold from 2006 to 2010 to more than $50B worldwide [17].

The nucleus of semantic technologies consists of *language standards* and a *core processing infrastructure*. The former is covered by the OWL, the latter is build of software that is necessary in the runtime environment when operating a semantically aware application. This report aims at providing a state of the art for both components, the Web Ontology Language as well as suitable reasoning systems. In order to achieve this, this report will shortly review the trends and developments of OWL and related standards. Against this background an update and extension of a previously conducted system evaluation [47] is given. Finally, there will be an discussion of upcoming application requirements in order to provide deeper insights into ontology based technology issues.

2. OWL Today and Tomorrow

2.1. Reviewing OWL

As mentioned in our preceding evaluation [47], OWL is layered on top of other fundamental Web language standards, namely XML and RDF. XML is the underlying document format whereas RDF provides an abstract data model. On top of RDF is RDF Schema (RDFS for short), which adopts the basic fact-stating ability of RDF and provides lightweight class- and property-structuring capabilities [12]. OWL extends the expressiveness of RDFS by adding further language elements while remaining sound and complete at least for the fragments OWL Lite and OWL DL. Evolutionary OWL has its origins in Description Logics (DL) [38], a research field within Knowledge Representation which itself is a sub-area of Artificial Intelligence.

Historically OWL evolved from the merge of two predecessor proposals, namely OIL and DAML. The premise in the course of developing OWL was driven by interoperability with existing Web standards (e. g. RDFS), practical tractability with help of inference engines, and usefulness of language features esp. with respect to naive users [9]. However, at the time of specification little was known about concrete demands and requirements from users outside the KR research community. But recently, due to the official status of OWL as a W3C recommendation, practitioners in industry and academia, tool developers, and others adopted OWL for usage within real or research applications. With this level of experience a manifold user community sat together in order to discuss how OWL could be applied, adapted and extended to fulfill current and future application demands at the 2005 OWL Experiences and Directions workshop[2] co-located with ISWC in Galway, Ireland. The participants identified several issues of OWL that are problematic in applications. The following is a selection of the most important or most frequently listed issues:

i) There are language restrictions which prevents from expressing some realities that have been rendered unnecessary by recent theoretical advances in logic-based

[2]Workshop Web site at `http://www.mindswap.org/2005/OWLWorkshop/`

Knowledge Representation.

ii) The layering on top of RDFS charges some extra technical burden on serialization and interoperability on OWL. In addition, due to its metamodelling architecture the RDFS semantics is not completely congruent to the OWL semantics (so called layering problem, see [52, 36] for example).

iii) There are still missing blocks in order to easily build an OWL aware application by plugging standard components together as required. For instance, there is still no official OWL query language or sufficiently complete communication protocol for standardized access to OWL inference engines.

iv) There is demand for coupling OWL with other representation frameworks such as rules, uncertainty, non-monotonicity, or spacial reasoning. In addition, there is need for having some sort of closed world reasoning in OWL.

2.2. OWL 1.1

As a result of the workshop, a revision of OWL-DL was proposed, which is called OWL 1.1. This still non-official extension is grounded on the above mentioned theoretical advances (item i) and has a well defined model-theoretic semantics. The logical basis ($\mathcal{SROIQ}$) is decidable and there yet exists a tableaux algorithm suitable for implementation [34].

The additional features of OWL 1.1 fall into five main categories:[3]

1. Syntactic sugar in order to make some commonly-stated things easier to say. These are `DisjointUnion`, `DisjointClasses`, and `EquivalentClasses`.

2. New Description Logic constructs. In particular these are qualified cardinality restrictions, local reflexivity restrictions, reflexive, irreflexive, anti-symmetric properties, disjoint properties, and property chain inclusion axioms.

[3]See `http://owl1-1.cs.manchester.ac.uk/Overview.html` for a more detailed description.

OWL DL: $\mathcal{SHOIN}(\mathbf{D})$ with $(n \geq 0)$ concrete domains $\mathbf{D}$ and GCIs

$$\{\underbrace{A, \top, r, r^+, C \sqcap D, C \sqcup D, \neg C, \exists r.C, \forall r.C}_{\mathcal{S}}, \underbrace{r \sqsubseteq s}_{\mathcal{H}}, \underbrace{\{i_1, \ldots, i_n\}}_{\mathcal{O}} \cdot \underbrace{r^{-1}}_{\mathcal{I}}, \underbrace{\geq n\ r, \leq n\ r}_{\mathcal{N}}\}$$

OWL 1.1: $\mathcal{SROIQ}(\mathbf{D}^+)$ which is all of OWL DL plus qualified number restrictions $\mathcal{Q}$, local reflexivity restrictions for simple properties; reflexive, irreflexive, and anti-symmetric flags for simple properties; disjointness of simple properties; and regular property inclusion axioms. (Note that some of them do not have a abstract syntax)

$$\{\underbrace{r \sqsubseteq s, r \circ s, U, \exists r.\mathsf{Self}}_{\mathcal{R}}, \underbrace{\geq n\ r\ C, \leq n\ r\ C}_{\mathcal{Q}}\}$$

Figure 1: Expressivity of OWL DL and OWL 1.1

3. Expanded datatype expressiveness for defining new datatypes. For instance, by restricting the domain of existing datatypes.

4. Meta-modeling constructs such that an identifier can be used as any or all of an individual, a class, or a property (often referred to as *punning*).

5. Semantic-free comments which can be interspersed throughout all expressions or sub-expressions of OWL ontologies.

Figure 1 sumarizes the language features of OWL 1.1. The implementors of the major Semantic Web reasoners, namely RacerPro, FaCT++, Pellet and Cerebra expressed a committment to support OWL 1.1 in the near future.

Regarding the layering problem with RDFS (issue ii) the community seems to shift towards a pure XML-based format for storing and exchanging OWL ontologies in the future. There is still a RDF syntax proposal[4] which, however, is defined by a transformation from the XML Exchange Syntax[5]. The latter also is intended to form the core of the new release of the Description Logic Interface (issue iii). The DIG 2.0 Interface aims at providing an implementation-neutral mechanism for accessing Description Logic reasoner functionality for all of OWL 1.1. At a high level the interface consists of XML

[4]See `http://owl1-1.cs.manchester.ac.uk/RDFsyntax.html`

[5]See `http://owl1-1.cs.manchester.ac.uk/XMLsyntax.html`

messages sent to the reasoner over HTTP connections, with the reasoner responding as appropriate. A preliminary description of DIG 2.0 [6] as well as its proposed extensions will be presented at the 2nd OWL Experience Workshop in November 2006. Current development efforts for DIG extensions cover access to previously told ontology information, retraction of ontology axioms, various non-standard inferences services as well as a conjunctive query language.

There is not much progress with respect to issue iv from above. There are no mature proposals how to combine uncertainty of spacial reasoning tightly with OWL. Except for rules, which have become manifest in the SWRL, a Semantic Web Rule Language Combining OWL and RuleML [37]. Some sort of local closed world reasoning have been integrated into RacerPro's new query language (nRQL) [26] but is not supported elsewhere.

3. Update and Refinement of Previous Evaluation

This section describes an update and significant extensions of a previous OWL resoner survey [47]. Just as the existing evaluation it will use the same (but slightly updated) evaluation criteria:

Language conformity: In order to serve as a reasoning component for the Semantic Web the language conformity according to the existing official OWL specification is an important evaluation criterion. In addition we check which OWL 1.1 language constructs a system is able to handle (compare with Figure 1).

Correctness: Soundness and completeness of the systems as given by the developers and on the an empirical base using our extended test samples.

Efficiency: Runtime and resource consumption of a few realistic ontologies as well as artificially compiled samples with different complexity.

Interface capabilities: Listing of system interface capabilities like interactive communication vs. batch-processing, support for loading URLs, programming interface, client-server architecture, etc. We also mention support of communication protocols such as the DIG 1.1 interface [8].

Inference services: Rating of the offered system services and system handling. Non-standard services will be mentioned when available.

3.1. Systems

In contrast to the previously set of systems as of September 2004 we have dropped to benchmark FaCT[6] as well as BOR[7] since both engines are no longer under active development and support only a fraction of OWL. Not surprisingly, the remaining tableaux-based systems FaCT++ [62], RacerPro [24], and Pellet [58] have evolved since then. In particular, FaCT++ is now available as release 1.1.3 (via approx. five intermediate

[6] `http://www.cs.man.ac.uk/~horrocks/FaCT/`

[7] `http://www.ontotext.com/bor/`

versions). RacerPro now is available in version 1.9.0, which is the second major release of the commercial version of Racer. Pellet latest available version is 1.3 with two non-beta releases since mid 2004. In addition, we have kept Hoolet (whose underlying prover vampire has not changed) as a well performing pure first order reasoner for comparison only. The only new inference engine we have added to our evaluation is KAON2 [51], which utilizes a disjunctive Datalog approach.

3.1.1. FaCT++

Developer. Dmitry Tsarkov, Medical Informatics Group, School of Computer Science, The University of Manchester, Oxford Road, Manchester, M13 9PL, UK, `http://owl.man.ac.uk/factplusplus/`

System Description. FaCT++ started as C++ re-implementation of the DL reasoner FaCT at the University of Manchester during the IST project WonderWeb [60]. It uses most of the established and highly optimized FaCT algorithms, but with a different internal architecture. During implementation new optimizations were also introduced (e. g. enhanced absorption techniques [61]) and some new features were added. The current version (release 1.1.3 from 04/2006) is stated to support $\mathcal{SHOIQ}(\mathbf{D}^-)$ and can be downloaded as executable together with sources from the project Web site [59]. The system is run via scripts producing textual output about inference results and prover statistics.

OWL Language Conformity. FaCT++ supports OWL DL with ABox and nominal reasoning resp. support. Since version 1.1.0 FaCT++ supports complex restrictions for Integer and String datatypes. It currently does only support a fraction of the role language features ($\mathcal{R}$) of OWL 1.1. Several bug fixes made FaCT++ much more stable and slightly faster since the last evaluation.

Correctness. FaCT++ is stated to be correct and sound but fails on four out of 30 sample tests.

Efficiency. FaCT++ performs extremely well within all of our test cases.

Interface. FaCT++ has no native OWL import – OWL ontologies have to be converted into FaCT++ syntax. An online conversion is available at [3]. The native FaCT++ syntax is similar but not compatible to the KRSS [55] standard and slightly different from the FaCT syntax. In addition, FaCT++ can create FOL problems for subsumption and satisfiability checking in the $\mathcal{SHOIQ}$ logic using a standard syntax (TPTP) that can be read by most first order theorem provers.
FaCT++ can be used either as a standalone software component, a DIG-enabled reasoner, or as a servlet implementing the HTTP DIG interface.

Inference Services. FaCT++ provides standard TBox reasoning tasks like subsumption and consistency checking as well as taxonomy construction. It also performs instance classification. Currently FaCT++ does not support any kind of query language. A unique feature of FaCT++ is given by the set of options which allow for detailed tuning of the internal reasoning process.
FaCT++ is one of the most recent and active developments. However, its current architecture is that of a traditional prover with limited interactive client access and no retraction or incremental reasoning features.

3.1.2. RacerPro

Developer. Volker Haarslev, Concordia University, 1455 de Maisonneuve Blvd. W., Montreal, Quebec H3G 1M8, Canada and Ralf Möller, Hamburg University of Science and Technology Software, Technology, and Systems (STS), Harburger Schloßstraße 20, 21079 Hamburg, Germany. `http://www.sts.tu-harburg.de/~r.f.moeller/racer/`

System Description. RacerPro implements a tableau calculus supporting multiple T- and ABoxes [24]. RacerPro incorporates all optimization techniques of FaCT [32] as well as some others for dealing with number restrictions and ABoxes. The current release 1.9.0 is the further developed commercial version of Racer version 1.7.24 from

Racer Systems[8]. RacerPro is avaliable as an executable server for Linux, Windows, and MacOS X. The system can be used either through a command line interface or via TCP or HTTP access using the client-server paradigm.

OWL Language Conformity. The RacerPro system implements the description logic $\mathcal{ALCQHI}_{\mathcal{R}+}(\mathbf{D}^-)$. In concrete, RacerPro can reason about OWL Lite knowledge bases, as well as OWL DL with approximations for nominals, together with some algebraic reasoning beyond the scope of OWL. Nominals in class definitions are approximated in a way which provides sound but incomplete reasoning. Within the current version RacerPro also allows to switch the unique name assumption (UNA) on or off (in contrast to earlier versions). RacerPro is able to reason with datatypes of type String, Integer, and Real. Similar to FaCT++ it currently does not support the newly introduced role expressions of OWL 1.1.

Correctness. No incorrect answers were found (there was one failure to conclude a datatype incoherence which is known not to be fully supported by version 1.9.0).

Efficiency. RacerPro was among the fastest systems except for very few tests within this hard TBox suite. Only one test case produced a timeout.

Interface. RacerPro is able of reading and writing KRSS, FaCT-XML, DAML+OIL and OWL syntax either locally or via HTTP access. Unfortunately OWL is serialized in a way tailored to the OilEd ontology editor [4] which does not conform to the official language specification. In addition, there are some bugs wrt. the OWL RDF import such as an error on reading empty intersections, unions, or one-of's or anonymous individuals. Using RacerPro as a server allows for TCP access on all available service features. Prototypical interface implementations for Java, CommonLisp and C++ are available. Another interactive communication possibility is the DIG 1.1 interface [8] which itself uses HTTP. RacerPro provides a facility to "dump" a server state into a file and to

[8] `http://www.racer-systems.com/`

restore the state from reloading the file.

Inference Services. Beyond standard reasoning task RacerPro offers quite a variety of different services especially for ABox retrieval. RacerPro allows for retraction of most of the TBox statements and ABox assertions as long as they are retracted using the identical syntax of their initial definition. In addition, RacerPro provides an ABox publish-subscribe mechanism to let users "subscribe" to an instance retrieval query which answers with a result list each time an upcoming ABox assertion matches. A recent extension of RacerPro is its new ABox query language nRQL [26] (new Racer Query Language) that can be used to provide access to extensionally specified information in ABoxes, which goes beyond standard ABox services. In addition, RacerPro supports so called substrates, which are conjunct representations and reasoning features e. g. for spatial information.
RacerPro offers broad and flexible interfaces and inference services. Nevertheless, there are some minor implementation flaws which result in problems/unexpected outcomes when dealing with multiple TBoxes or retracting given TBox statements.

3.1.3. Pellet

Developer. Evren Sirin and Bijan Parsia, Maryland Information and Network Dynamics Laboratory Semantic Web Agents Project, University of Maryland, 8400 Baltimore Avenue, Maryland 20740, `http://www.mindswap.org/2003/pellet/`

System Description. Pellet is a reasoner based on well-known tableau algorithms for T- and ABox reasoning. It is developed at the University of Maryland intended for reasoning about Web Services. Pellet is completely written in Java and available as as Java class archive together with its source code from [22]. However, the source code is barely commented. The latest available version 1.3 is told to be sound and complete by incorporating the recently developed decision procedure for $\mathcal{SHOIQ}$ [39] (the expressivity of OWL-DL plus qualified cardinality restrictions in DL terminology). An online demo can be found on the project Web page.

OWL Language Conformity. Pellet supports the DL known as $\mathcal{SHOIQ}$(D) (in contrast to $\mathcal{SH}\{\mathcal{I}|\mathcal{O}\}\mathcal{N}$(D) within our earlier evaluation) which corresponds to OWL DL with user-defined datatypes [58]. Pellet is the only system which claims to support the full XML Schema datatypes.

Correctness. Pellet failed on two of 30 test cases (and produced a memout in a further case). These two test included neither inverse roles nor nominals. As the previous systems the additional role expressions of OWL 1.1 are not supported yet.

Efficiency. Pellet performs reasonable well. It has significantly improved in comparison to our previous benchmarks. Unfortunately, it was not able to process the largest ontology of our evaluation (GALEN) because of an out-of-memory error.

Interface. Pellet can read OWL syntax and is run via command line. In addition, the system can be run against the official OWL test cases via HTTP access using the corresponding main manifest file. It offers support for DIG 1.1 in terms of a DIG server and is a component of the SWOOP [43] OWL editor. It comes with a reasoner API for Jena [13] and the OWL API [7].

Inference Services. The available Pellet inference services are detection of unsatisfiable classes, checking for entailed statements and building the class taxonomy. It also provides ABox realization by showing classified individuals in the class hierarchy. An exceptional property of Pellet is its ontology analysis and repair feature trying to convert OWL Full ontologies into OWL DL. It also bears some non-standard debugging features [54] as well as ontology partitioning functionality based on the e-connection calculus [20]. Pellet also supports the conjunctive query languages SPARQL and RDQL.

3.1.4. KAON2

Developer. Boris Motik, Information Management Group, University of Manchester, 2.46 Kilburn Building, Oxford Road, MANCHESTER, M13 9PL, UK. `http://kaon2.`

semanticweb.org/

System Description. KAON2 is an infrastructure for managing OWL-DL, SWRL, and F-Logic [45] ontologies. Reasoning is implemented by novel algorithms which reduce a knowledge base to a disjunctive Datalog program. The algorithm for reducing description logic knowledge bases into disjunctive datalog is based on basic superposition [2]. The resulting datalog program is solved while making use of well known deductive DB technology such as the magic set transformation algorithm [16]. KAON2 is available as a precompiled binary Java distribution and is free of charge for research and academic purposes. A commercial version has to be licensed from Ontoprise GmbH.[9] After a release of KAON2 at the end of 2005 the next version was released in August 2006 which has been used for our evaluation. Since then many 'silent' updates have been released without any documentation or release notes. According to e-mail communication with Boris Motik KAON2 now better supports datatypes and has a more mature F-Logic handling. Future work will cover the integration of negation-as-failure.

OWL Language Conformity. KAON2 supports the $\mathcal{SHIQ}$(D) subset of OWL DL. This includes all features of OWL DL apart from nominals. Concerning OWL 1.1 it supports qualified number restrictions but currently no property chains nor anti-symmetry, reflexity, etc.

Correctness. Reasoning within KAON2 is provable sound and complete [41]. Accordingly, no incorrect answers found within the test cases KAON2 supports.

Efficiency. KAON2 is known to perform very well on large ABoxes. Within our test set of difficult but small TBox cases it either answers immediately or produces a memout (or timeout depending on the memory - CPU performance ratio). Especially in the presence of cardinality restrictions with cardinalities greater or equal to two KAON2 often runs into a memout error.

[9] http://www.ontoprise.de/

Interface. There is DIG 1.1 support within KAON2. It can also be used as a stand-alone server or library using RMI within Java. In addition it contains a module for extracting ontology instances from relational databases.

Inference Services. Due to the fact that KAON2 aims at reasoning with large amounts of individuals [50], it is short on elaborated TBox services. It supports the basic set of services via DIG 1.1 or directly via its API. In addition it supports SPARQL for answering conjunctive queries.

3.1.5. First Order Theorem Prover Hoolet

Although not a tableaux-based reasoner, Hoolet uses an promising alternative processing architecture. This makes it interesting to benchmark Hoolet against the other systems.

Developer. Dmitry Tsarkov, Medical Informatics Group, School of Computer Science, The University of Manchester, Oxford Road, Manchester, M13 9PL, UK, `http://owl.man.ac.uk/hoolet/`

System Description. Hoolet is an implementation of an OWL-DL reasoner that uses a first order prover [63]. The ontology is translated to a collection of FOL axioms (in an obvious way based on the OWL semantics) and this collection of axioms is then given to a first order prover for consistency checking. Hoolet uses TPTP as intermediate format. TPTP is a standardized format used for benchmarking first order theorem provers. Hoolet uses the first order theorem prover Vampire [56] (version 6). The translation - prover bundle has not evolved since our last evaluation.

OWL Language Conformity. Hoolet is implemented using the WonderWeb OWL API [7] for parsing and processing OWL, and the Vampire prover for reasoning purposes. Other reasoners could also be used as long as they support the TPTP format. Since $\mathcal{SHOIN}$ is a fragment of FOL Hoolet is able to cope with all language constructs of OWL DL.

Correctness. The underlying prover vampire is known to be sound but incomplete. Our tests produced no errors. There are a couple of timeouts and memouts however.

Efficiency. The underlying reasoner Vampire has proven to be one of the fastest systems for FOL. However, this does not necessarily imply to be likewise fast on the OWL DL fragment of FOL. In fact, when using a naive translation Vampire seems to be slow (see [63] for more detailed discussion). However, more smart translations with some kind of preprocessing (e.g. axiom absorption or leaving out irrelevant axioms) promises to speed up the reasoning process.

Interface. Hoolet is a OWL to TPTP translation process using Vampire rather than an own system implementation. For more convenient usage Hoolet provides a Java GUI for translation and querying OWL ontologies. However, an interactive and flexible interface is not present.

Inference Services. The GUI of Hoolet allows for the standard TBox and ABox queries. Expressive queries (e.g. with nested statements) are not expressible.

3.2. Test Cases

The test cases underlying our evaluation are not randomly selected. Instead, they try to meter the correctness of reasoning engines with respect to inference problems of selected language features. In this respect, our analysis fits into the history of system evaluations more or less initiated by Heinsohn et. al. [30] and continued by system comparisons at various DL workshops.

Note however, that our test cases are no more than spot tests which are intentionally designed to be hard to solve but small in size. In fact, our test cases typically consist of less than a dozen classes, properties, or individuals. Obviously, such an empirical procedure can not prove a system to be correct but incorrect in case of detecting an error.

The developed test suite only contains hard TBox tests, except for nominals (which enroll ABox reasoning into the TBox to some extend) and some special test on ABox Open World Reasoning (OWA). The previous evaluation contained 12 test cases (no. 1a to 8) which are also included into this update. In addition, we have extended the test set to 34 tests which now cover other aspects such as nominals and datatypes.

Clearly, this test suite evaluates only one, but important facet of OWL inference engines, namely correctness with help of difficult TBox tests. There are other aspects like ABox reasoning and scalability, which are likewise important for real-world applications. These issues are investigated in one of our other reports [66].

As mentioned before, the various test cases check for different inference features which are enforced due to the usage of specific language constructs or combinations thereof. A detailed description of the particular inference or language features for each of the test cases is beyond the scope of this report and would require detailed background knowledge in the field of logic-based knowledge representation as well as proof theory.

Experienced readers which are familiar with the abstract DL language notation [1] are referred to Appendix A. There, all 34 test cases are listed with their associated entailment query in DL abstract syntax. Futhermore, their corresponding RDF/XML serialization is accessible at URL: `http://www.informatik.uni-ulm.de/ki/Liebig/reasoner-eval/`*testnumber*`.owl`.

In addition, each test case is briefly characterized by a short description and an expressivity classification in the result table of sub-section 3.3. The expressivity characterization shows the corresponding OWL language fragment as well as in the more fine graded DL language abbreviation schema. One can use the DL complexity navigator Web site [70] in order to get an idea about the worst case complexity of the corresponding entailement problem. Beyond that our result table lists the number of classes (C), object properties (P), datatype properties (D), and individuals (I) for each test case separately.

3.3. Testing Results

The following describes the results of the extended TBox test cases mentioned in the previous section. They are given with help of a table and an associated discussion of selected results. There are result columns for each of the described systems in subsection 3.1. In addition, we have also listed the results of two predecessor systems, namely Racer version 1.7.24 (the last freely available release of mid 2004) and Pellet version 1.3 beta (of 09/2005) in order to make the development progress of the field explicitly visible. There are also some notes on the predecessor releases of FaCT++.

The tests were performed on a SuSE Linux 10 computer with an 3GHz Intel processor. The main memory limit was set to 600MB for Java (Pellet & KAON2). The other systems were run on their default because of missing options. If a system ran out of memory this is indicated by a "memout". The time limit was 10 minutes and resulted in a "timeout" in case of non-termination within this limit. Note that there is no hard distinction between memory- and time-out. Which one comes first sometimes is a matter of the memory / cpu performance ratio. A "+" indicates that the system passed the test. Round brackets are a sign of some special remark typically given as a table footnote. If a system didn't respond within a second the processing time is given in round brackets. A "–" indicates a wrong result. A "**X**" indicates that the system doesn't support the language of the test case. Results in square brackets indicate that the system uses approximate reasoning for this test case (e.g. in the case of nominals). "err" indicates that the system aborts with a runtime error. Test case no. 11 includes three small test which are transmitted via OWL RDF/XML syntax (first line) as well as via DIG 1.1 (second line).

3.3.1. Discussion of Results

The first six tests (**1a to 3b**) are of similar type in the sense that they check for cardinality merging abilities within different expressive language fragments. KAON2 shows the expected problems with cardinalities for four out of six cases. A similar behavior is shown by Hoolet, which indicates that DL tableaux approaches can handle

this kind of non-determinism more efficiently. However, there are limits which can be seen by one RacerPro timeout and significant runtime by FaCT++ and Pellet form some of the cases.

Test **4 to 6** focus on blocking abilities with or without inverse properties. They show one timeout for Hoolet as well as one for an earlier version of Pellet.

Nominal test **7** were solved by all systems which natively support this language feature.

Open world test **8** was solved by all systems correctly.

The three property filler merging tests **9 to 10b** identify one wrong answer for Pellet (case 9). An earlier version of FaCT++ failed also. Due to non-determinism all except FaCT++ require significant time (up to timeout) to solve 10a/b. Interestingly, Pellet and KAON2 show an unpredictable behavior for 10a/b. The time range for these tests vary from some seconds up to timeout from run to run.

Test **11** is a combined syntax/special test case. It checks whether the systems handle empty unions, intersections, or enumerations logically correctly. This test comes with two flavors: via an RDF/XML (upper row) and DIG 1.1 (lower row) import. The outcome of this test was surprisingly worse. No system was able to handle all of the three sub-test via one of the syntax serializations logically correct. FaCT++, RacerPro, and KAON2 even returned with parsing errors.

The individual merging test **13** was solved by the actual releases of the engines which natively support nominals (FaCT++, Pellet, Hoolet). Earlier versions of FaCT++ and Pellet failed on this test. Interestingly, the proof of the inverse subsumption query needed 75s to answer by Hoolet. Case **14** is a related nominal test which passed all nominal-enabled reasoner.

A test focusing on reasoning with inverse roles is number **15** and was solved by all systems.

Engines supporting nominals were also able to solve test **16** (others were not).

Detecting an inconsistent ontology within test **17** resulted in an error for FaCT++ and failed for Pellet 1.3 beta.

Very surprisingly, FaCT++ failed to reason about $A \sqcup \neg A \equiv \top$ in test **18**. For KAON2

we were forced to introduce an auxiliary individual in order to check for the entailment.

Test **19** is a syntax check whether there are complex properties which are not allowed within at-most cardinality restrictions and a transitive property which can not be a sub-property of a functional property. Otherwise the underlying logic is loosing its decidability. Unfortunately, FaCT++ as well as Pellet do not check for this.

The infinite model test of case **20** is passed by all actual system releases.

The datatype property test **21** build up a datatype property hierarchy, assigns some fillers and checks whether the system assume that datatype properties are functional per se (which they are not). FaCT++ however, quits with an error during processing and RacerPro can only solve a part of the query. The related datatype test **22** defines an unsatisfiable class due to conflicting range restrictions of a datatype ($\geq 0 \wedge \leq -1$). RacerPro and Hoolet fail on this test. In addition, we were not able to figure out whether FaCT++ concludes this unsatisfiability. The tested version of KAON2 had not datatype support (the newest release does).

Another surprise was that FaCT++ fails on the very simple partitioning test **23**.

The Open World nominal test **25** was solved by all systems even when they simulate nominals with classes.

A kind of nominal merging within test **26** was passed by all nominal-aware systems.

Test case **27** was inspired by a posting of Evren Sirin on the OWL mailing list. It aims at an sub-property entailment with help of nominals. We had problems with most systems to figure out whether they are able to conclude this. The equivalence of the two auxiliary classes T1 and T2 suggest, that the systems FaCT++, RacerPro, and Pellet conclude the properties as equivalent.

The and-branching test **28** is a check on efficient propagation of filler restrictions and non-determinism. Except for Hoolet which quits with an memout error all systems were able to solve it. Here, Pellet was the slowest system, KAON2 and FaCT++ the fastest.

The test **29a/b** is again a cardinality merging problem (with non-determinism). Surprisingly, FaCT++ was not able to solve **29b** (it solved all other more complex merging problems however). Even Hoolet was able to solve them while taking some time.

Test **30** is different from all of the other tests. Instead of checking for a special feature it tries to use all features within one small example. This example consists of a cons-like list representation using nominals, inverse roles, GCI's and sophisticated restrictions as well as anonymous individuals. No system, except FaCT++, was able to handle this test case. Even FaCT++ needed 28s to classify the test case. RacerPro was faster but does not support pure nominals, which are a key of this example. Pellet and Hoolet ran into a memout error.

3.3.2. Conclusion

The overall outcome of this evaluation is somehow disappointing. No system, except RacerPro and KAON2, was able to correctly solve (in case of termination) at least those tests which lay within the language fragment they claim to support in full. Even worse, the detected failures were unpredictable across each system. In other words, if users consider correctness as an important factor they can not trust these systems for usage in real-world applications.

To some extend KAON2 and Hoolet are not application ready since they fail very often with out of memory errors or require significant processing time for language constructs which are typically in real-world models such as cardinality restrictions. KAON2 however never produced a serious failure but does not support nominals. Pellet and FaCT++ do have some serious bugs which result in incorrect answers but support the most expressive language fragment. RacerPro failed only for one special datatype problem (which are not fully supported) but cannot handle nominals natively.

	Expressivity	**Description**	C	P	D	I	**FaCT++ 1.1.3**	**Racer 1.7.24**	**Racer 1.9.0**	**Pellet 1.3b**	**Pellet 1.3**	**KAON2 08/06**	**Hoolet vampire6**
1a	OWL DL $\mathcal{ALCHIN}$	cardinality merging (all satisfiable)	6	7	0	0	+	+	+	+	+	memout	timeout
1b	OWL DL $\mathcal{ALCHIN}$	cardinality merging (one unsatisfiable)	6	7	0	0	+ (7 sec)	timeout	timeout	+	+	memout	timeout
2a	OWL DL $\mathcal{ALCHN}$	cardinality merging (all satisfiable)	8	9	0	0	+	+	+	+ (17 sec)	+ (3 sec)	memout	timeout
2b	OWL DL $\mathcal{ALCHN}$	cardinality merging (one unsatisfiable)	8	9	0	0	+ (120 sec)	+	+	+	+	memout	timeout
3a	OWL DL $\mathcal{ALCN}$	cardinality merging (all satisfiable)	7	1	0	0	+	+	+	+	+	+	+
3b	OWL DL $\mathcal{ALCN}$	cardinality merging (one unsatisfiable)	7	1	0	0	+	+	+	+	+	+	+
4	OWL Lite $\mathcal{ALIF}$	cycle/inv. blocking (all satisfiable)	4	2	0	0	+	+	+	timeout	+	+	timeout
5	OWL Lite $\mathcal{ALCIF}$	cycle/inv. blocking (one unsatisfiable)	5	2	0	0	+	+	+	+	+	+	+
6	OWL Lite $\mathcal{ALCIF}$	inverse blocking (one unsatisfiable)	7	3	0	0	+	+	+	+	+	+	+
7	OWL DL $\mathcal{ALCOF}$	nominal/cardinality (all satisfiable)	3	1	0	1	+	+	[-]	+	+	**X**	+
8	OWL Lite $\mathcal{ALCF}$	OWA (all satisfiable)	6	2	0	4	+	+	+	+	+	+	+
9	OWL Lite $\mathcal{ALCHF}$	functional role merging (all satisfiable)	7	6	0	0	+[1]	+	+	-	-	+	+
10a	OWL Lite $\mathcal{ALCN}$	role filler merging (one unsatisfiable)	19	1	0	0	+	+ (160 sec)	+ (110 sec)[2]	(+) (10 sec)[3]	(+)[4]	(+)[3]	timeout[8]
10b	OWL Lite $\mathcal{ALCN}$	role filler merging (all satisfiable)	19	1	0	0	+	+ (150 sec)	(+) (110 sec)[1]	(+)[4]	(+)[3]	+	???
11	OWL Lite/DL $\mathcal{AL}/\mathcal{ALC}/\mathcal{ALO}$	empty intersec. (satisf.) union/oneOf (unsatisf.)	3	0	0	0	err/err/- +/-/-	+/+/[+] +/+/[+]	err/err/[err] +/+/[+]	-/+/+ -/+/+	-/+/+ -/+/+	+/+/– err/+/–	— —[6]
13	OWL DL $\mathcal{ALCON}$	individual merging (class subsumption)	4	1	0	5	+[1]	[-][12]	[-]	err	+	**X**	+[9]
14	OWL DL $\mathcal{ALCO}$	nominals (all satisfiable)	6	1	0	4	+	[-]	[-]	+	+	**X**	+

	Expressivity	**Description**	C	P	D	I	**FaCT++ 1.1.3**	**Racer 1.7.24**	**Racer 1.9.0**	**Pellet 1.3b**	**Pellet 1.3**	**KAON2 08/06**	**Hoolet vampire6**
15	OWL Lite $\mathcal{SHIF}$	transitivity/inverse role test (one unsatisfiable)	9	6	0	0	+	+	+	+	+	+	+
16	OWL DL $\mathcal{ALCOF}$	nominal/cardinality (all satisfiable)	4	1	0	1	+	[-]	[-]	+	+	**X**	+
17	OWL Lite $\mathcal{ALC}$	inconsistent ontology (all unsatisfiable)	2	0	0	0	err	+	+	-	+	+	+
18	OWL Lite $\mathcal{ALC}$	union test ($A \sqcup \neg A \equiv \top$)	2	0	0	0	-	+	+	+	+	(+)[10]	+
19	OWL Lite $\mathcal{SHF}$	syntax test (card. restr. over complex roles)	1	4	0	0	-	+	+	-	-	+	**X**[7]
20	OWL Lite $\mathcal{SHIF}$	infinite model test (all unsatisfiable)	3	2	0	0	+	+	+	timeout	+	+	+
21	OWL DL $\mathcal{ALCHN}(D)$	datatype properties (all satisfiable)	3	0	3	1	err	[-]	(+)	+	+	**X**	+
22	OWL DL $\mathcal{ALE}(D)$	datatype properties (one unsatisfiable)	1	0	1	0	???	[-]	-	+	+	**X**	-
23	OWL DL[5] $\mathcal{ALC}$	partitioning test (class equivalence)	6	0	0	0	-	+	+	+	+	+	+
25	OWL DL $\mathcal{ALCHOF}$	open world test (case analysis)	4	4	0	6	+	[-][13]	[+]	+	+	**X**	+
26	OWL DL $\mathcal{ALCON}$	nominal mergin (class subsumption)	7	1	0	11	+	[-]	[-]	err	+	**X**	+
27	OWL DL $\mathcal{ALCOF}$	sub-property reasoning	3	2	0	3	-[11]	[-]	[-]	(+)[11]	(+)[11]	**X**	-[11]
28	OWL Lite $\mathcal{ALC}$	and-branching test (all satisfiable)	33	7	0	0	+	+ (36 sec)	+ (16s)	+ (88s)	+ (60s)	+	memout
29a	OWL DL[5] $\mathcal{ALCHIF}$	cardinality merging (all satisfiable)	12	6	0	0	+	+	+	+	+	+	+ (70sec)
29b	OWL DL[5] $\mathcal{ALCHIF}$	cardinality merging (one unsatisfiable)	12	6	0	0	-	+	+	+	+	+	+ (8sec)
30	OWL DL $\mathcal{SHOIF}$	list representation	30	8	0	40	+ (28sec)	[-]	[+]	memout	memout	**X**	memout

Listing of the footnotes used within the result table:

1) Version 1.1.2 failed

2) Nearly two-third of the time has been spend to lisp garbage collection

3) Measured time varies from 10 sec. up to timeout

4) Measured time varies from less than one second up to timeout

5) Can be reformulated as OWL Lite

6) Depends on OWL API converter library

7) Based on FOL prover; converter needs to check for this

8) Constant memory consumption

9) The inverse subsumption query needs 75s to answer

10) When using an auxiliary class $MyTop \equiv \top$ and with $i : MyTop$we can infer that $i : AnotA$

11) Difficult to test because sub-property reasoning is not a standard task (FaCT++ and Racer do not show correct sub-property relationship) (Pellet doesn't even show property hierarchy) $T1 \equiv T2$ implies property equivalence for FaCT++, Racer, and Pellet

12) Produces an error report during loading (malformed oneof expression)

13) Reports about incoherent ABox (parsing problem probably)

4. Technology Insights and Trends

This section of the report tries to provide some insights into current issues and trends in the field of ontology reasoning. Note, however, that the following is tailored to DL and closely related reasoning techniques. Related but incomplete systems which, for example, utilize a rule-based technique such as OWLIM [46], JENA [14], and others are not covered. In order to gain details about those approaches with respect to ABox performance please consult our other report [66].

4.1. Non-Standard Reasoning Services

Especially since ontologies are build, maintained, as well as used even by non-sophisticated users, there is need for services which support ontology engineers in this task. Such services are non-standard in the sense that they have a very special focus or are build on top of existing reasoning capabilities.

One of the most active sub-fields in non-standard reasoning aims at providing explaining and debugging support to users. An explaining component tries to provide an user understandable trace of conclusions for some standard reasoning service. There are currently explaining techniques for three of the most central reasoning services, namely class subsumption, class unsatisfiability, and instance classification.

An approach which utilizes a DL tableaux approach for explaining subsumption and unsatisfiability is given in [49]. It implements and extents previous work of Borgida et. al. [11] as an extension of the OntoTrack OWL authoring tool [48]. However, this approach currently can not handle nominals as well as inverse roles. A more expressive and different technique is presented in [44]. Here, an axiom pinpointing approach is used to identify those axioms which are responsible for a certain entailment. The problem with the latter technique is that it is more coarse in the identification of sub-expressions in comparison to the former. Recent work claims to be able to provide more fine-grained justifications [42]. These techniques are also employed to explain the reason for unsatisfiable classes and implemented in the OWL editor SWOOP [43]. A major drawback of the axiom pinpointing approach is that it does not scale very well. It uses

a DL reasoner as a black box (oracle) for certain questions. Due to missing internal information of axiom dependencies it has to take the whole ontology into account which makes it practically useless for large ontologies.

Another kind of non-standard inference service are those who support users in building new classes. Sonic[10] for instance is as prototype system that implements two inferences, namely the least common subsumer (lcs) and the approximation inference [64]. A least common subsumer is of benefit in the case of constructing an ontology bottom-up. Here, instead of directly defining a new class, the knowledge engineer selects several typical classes as examples, which are then automatically generalized into a new class description by the lcs service. A second non-standard service is the approximation of a class typically in a less expressive language (i. e. not supporting disjunction).

Very likely such kind of non-standard inference services are found in many OWL authoring tools in the near future. Some of the services above are currently being defined as extensions to the new DIG 2.0 specification which allows to access them within different applications more easily.

Future work in the field of non-standard inferences will also address the explanation of non-entailments. For instance, users may be interested in an explanation showing why a class is not subsumed by another one, or why an individual is not an instance of a certain class.

4.2. Experiences and Practical Hints

Reasoning with OWL ontologies is of high worst-case complexity. OWL DL's worst-case complexity is NExpTime but actual tableaux based DL reasoners employ an algorithm of at least 2NExpTime complexity. Although this worst-case does not show up within most practical applications, our experiences have shown that one can incidentally create a particular feature combination which significantly can degrade performance. This is due to feature interactions which some times force inference engines to switch certain optimizations off. The usage of inverse properties is such a language feature which

[10] `http://wwwtcs.inf.tu-dresden.de/~sonic/`

disables some very efficient caching method in tableaux based systems. Even if there is some progress to apply caching in presence of inverse roles in some cases [18] the developers of the RacerPro reasoner therefore recommend to avoid inverse properties whenever possible [27].

Beyond that, cyclic class definitions require special blocking techniques which add burden to the tableaux algorithm in terms of extra checks during reasoning. Together with inverse properties a cyclic definition requires even more blocking checks. Another language feature with influence on reasoning performance are nominals. This is mainly due to the fact that they are one of the newest features of DL based languages. Research on optimization techniques with respect to nominals has just started.

However, personal experiences have shown that one has to use the above mentioned features in class or property definitions in order to slow down reasoning. Just adding those features to an ontology without usage in some of the defining axioms is without consequence. On the other hand, just one occurrence potentially can have significant effects, because it will switch of optimizations for all definitions which directly or indirectly refer to that statement.

4.3. Dynamic Aspects of Ontologies

Users of OWL have often expressed their needs towards support for reasoning with fluctuating data. In other words, whereas the TBox typically remains constant, the ABox is subject to change frequently. However, current reasoning systems operate more or less in a batch-oriented fashion. Whenever the ontology changes they discard all inferences and start from scratch. A more efficient way for dealing with frequently changing ABoxes has recently been presented in [28]. This approach updates a previously computed tableau completion graph under syntactic addition and removal of ABox assertions for $\mathcal{SHOQ}$ and $\mathcal{SHIQ}$. Based on this work an optimization for conjunctive query answering has been developed in [29]. The proposed technique aims at reducing candidate variable bindings for queries when the ABox is updated.

Both approaches are prototypically implemented in Pellet and show an significant

performance speed up and promise to stimulate further research and adaption by other system developers.

4.4. Scalability

Scalability has always been and still is an issue for system developers [33]. However, during the 90s scalability has always been investigated with respect to the amount of TBox axioms (ABox reasoning was typically not supported at that time). As a result many highly efficient TBox optimizations are known and are implemented in almost all systems [35, 31]. The most important optimizations are lazy unfolding, dependency directed backtracking, absorption, and semantic branching. A cheap sound but incomplete syntactical subsumption check is the pseudo-model merging test which can significantly reduce the number of subsumption proofs [25].

Since TBox efficiency has been in the focus of research for many years, there has not been much progress in this area recently. An exception to this is an optimization for handling of nominals [57] for the tableaux decision procedure able to handle $\mathcal{SHOIQ}$ [39]. Interestingly, the latter has not shown to very effective within our nominal test case number 30.

Much more interesting results have been gained in the area of scalable ABox reasoning and query answering within the last years. In order to be able to deal with large sets of individuals developers tried to combine DL inference engines with Database systems (e. g. Instance Store [5], LAS [15]). Unfortunately those approaches restrict the expressiveness of the ABox axioms in a way which is not acceptable for real-world applications (for instance, property instantiations are not possible within Instance Store). With the appearance of KAON2, which utilizes Database technology within a sound and complete approach, more pressure were put on the tableaux based system developers. KAON2 is able to handle very large sets of individuals [50]. Until then a couple of efficiency increasing methods for tableaux systems have been implemented. Key techniques are indexing techniques and re-ordering of clauses within conjunctive queries [69]. A more detailed discussion of ABox reasoning performance can be found in our other report [66].

In addition, there are other approaches which could help to increase reasoning with individuals. For instance, in [19] an ABox reduction technique has been used to reduce the number of individuals a system has to reason about for consistency detection in $\mathcal{SHIN}$. Future work has to show whether similar techniques could be used for query processing.

To sum up, there are promising techniques for scalable TBox as well as ABox reasoning. However, current real-world ontologies are not very expressive [65] in order to gain deeper insights about practical scalability. Artificial benchmarks have shown to be not well suited to serve as realistic test cases [67].

4.5. Others

As mentioned in section 2.1 users would like to combine DL based reasoning with other representation frameworks. A step in this direction is offered by RacerPro which supports so called substrates as add-ons to the base DL. A substrate of RacerPro 1.9.0 is the Region Connection Calculus (RCC), a first order axiomatic theory for qualitative spatial reasoning. With help of this calculus users can combine qualitative spatial information with traditional DL representations. Early prototypical application in the context of digital city maps [68] and anatomy [10] still do not allow for a substantial review.

Another interesting idea is the automatic partitioning of ontologies [21]. This approach is based on $\mathcal{E}$-connections which can be use to break up an ontology into components which do not depend on information from any other component. This allows for collaborative authoring as well as independent (parallel) reasoning.

References

[1] Franz Baader, Diego Calvanese, Deborah McGuinness, Daniele Nardi, and Peter Patel-Schneider, editors. *The Description Logic Handbook*. Cambridge University Press, 2003.

[2] L. Bachmair, H. Ganzinger, C. Lynch, and W. Snyder. Basic paramodulation. *Journal Information and Computation*, 121(2):172–192, 1995.

[3] Sean Becherhofer and Raphael Volz. OWL Ontology Converter. `http://phoebus.cs.man.ac.uk:9999/OWL/Converter`.

[4] Sean Bechhofer, Ian Horrocks, Carole Goble, and Robert Stevens. OilEd: a Reasonable Ontology Editor for the Semantic Web. In *Proc. of the German conference on Artificial Intelligence, KI2001*, pages 396 – 408. Springer Verlag, LNAI Vol. 2174, September 2001.

[5] Sean Bechhofer, Ian Horrocks, and Daniele Turi. The OWL Instance Store: System description. In *Proc. of the Int. Conf. on Automated Deduction (CADE-20)*, volume 3632 of *LNCS*, pages 177–181, Tallinn, Estonia, July 2005.

[6] Sean Bechhofer, Thorsten Liebig, Marko Luther, Olaf Noppens, Peter Patel-Schneider, Boontawee Suntisrivaraporn, Anni-Yasmin Turhan, and Timo Weithöner. DIG 2.0 – Towards a flexible interface for Description Logic reasoners. In *Proc. of the OWL Experiences and Directions Workshop (OWLED'06) at the ISWC'06*, 2006. submitted.

[7] Sean Bechhofer, Phillip Lord, and Raphael Volz. Cooking the Semantic Web with the OWL API. In Dieter Fensel, Katia Sycara, and John Mylopoulos, editors, *Proc. of the 2003 International Semantic Web Conference (ISWC 2003)*, number 2870 in Lecture Notes in Computer Science, pages 659–675. Springer Verlag, 2003.

[8] Sean Bechhofer, Ralf Möller, and Peter Crowther. The DIG Description Logic

Interface. In *Proc. of International Workshop on Description Logics (DL2003)*, Rome, Italy, 2003.

[9] Sean Bechhofer, Frank van Harmelen, Jim Hendler, Ian Horrocks, Deborah McGuinness, Peter F. Patel-Schneider, and Lynn Andrea Stein. OWL Web Ontology Language Reference. W3C Recommendation, February 2004. `http://www.w3.org/TR/owl-ref/`.

[10] M. Boeker, D. Raufie, and S. Schulz. Deskriptions-Logik basierte rumlich-topologische Reprsentation anatomischer Strukturen mit dem Region Connection Calculus. In *Proc. of the 51. Jahrestagung der Deutschen Gesellschaft für Medizinische Informatik, Biometrie und Epidemiologie (GMDS)*, 2006.

[11] Alex Borgida, Enrico Franconi, and Ian Horrocks. Explaining $\mathcal{ALC}$ subsumption. In *Proc. of International Workshop on Description Logics (DL1999)*, Linköping, Sweden, 1999.

[12] Dan Brickley and R.V. Guha. RDF Vocabulary Description Language 1.0: RDF Schema. W3C Recommendation, February 2004. `http://www.w3.org/TR/rdf-schema/`.

[13] Jeremy J. Carroll, Ian Dickinson, Chris Dollin, Dave Reynolds, Andy Seaborne, and Kevin Wilkinson. Jena: Implementing the Semantic Web Recommendations. In *Proc. of the Thirteenth International World Wide Web Conference (WWW2004)*, pages 74–83, New York, NY, USA, May 2004.

[14] Jeremy J. Carroll, Ian Dickinson, Chris Dollin, Dave Reynolds, Andy Seaborne, and Kevin Wilkinson. Jena: Implementing the Semantic Web Recommendations. In Stuart I. Feldman, Mike Uretsky, Marc Najork, and Craig E. Wills, editors, *WWW (Alternate Track Papers & Posters)*, pages 74–83. ACM, 2004.

[15] Cui Ming Chen, Volker Haarslev, and JiaoYue Wang. LAS: Extending Racer by a Large Abox Store. In Horrocks et al. [40], pages 200–207.

[16] C. Cumbo, W. Faber, G. Greco, and N. Leone. Enhancing the magic-set method for disjunctive datalog programs. In *Proc. of the 20th Int. Confe. on Logic Programming (ICLP'04)*, pages 371–385, Saint-Malo, France, September 2004.

[17] Mills Davis. Semantic Wave 2006: Part-1. Technical report, Project10X, Washington, DC, USA, 2006.

[18] Y. Ding and V. Haarslev. Towards efficient reasoning for Description Logics with inverse roles. In Horrocks et al. [40], pages 208–215.

[19] Achille Fokoue, Aaron Kershenbaum, and Li Ma. $\mathcal{SHIN}$ ABox Reduction. In Parsia et al. [53], pages 135–142.

[20] Bernardo Cuenca Grau, Bijan Parsia, Evren Sirin, and Aditya Kalyanpur. Automatic Partitioning of OWL Ontologies using e-connections. In Horrocks et al. [40].

[21] Bernardo Cuenca Grau, Bijan Parsia, Evren Sirin, and Aditya Kalyanpur. Automatic partitioning of owl ontologies using e-connections. In Horrocks et al. [40], pages 128–135.

[22] Michael Grove. Pellet OWL Reasoner. `http://www.mindswap.org/2003/pellet/`, 2003.

[23] Volker Haarselv and Ralf Möller, editors. *Proc. of the Int. Workshop on Description Logics (DL'04)*, volume 104 of *CEUR*, Whistler, BC, Canada, June 2004.

[24] Volker Haarslev and Ralf Möller. RACER System Description. In *Proc. of the International Joint Conference on Automated Reasoning (IJCAR'2001)*, volume 2083 of *LNAI*, pages 701–706, Siena, Italy, 2001.

[25] Volker Haarslev, Ralf Möller, and Anni-Yasmin Turhan. Exploiting Pseudo Models for TBox and ABox Reasoning in Expressive Description Logics. In R. Goré, A. Leitsch, and T. Nipkow, editors, *International Joint Conference on Automated*

Reasoning, IJCAR'2001, June 18-23, Siena, Italy, pages 29–44. Springer-Verlag, 2001.

[26] Volker Haarslev, Ralf Möller, and Michael Wessel. Querying the Semantic Web with Racer + nRQL. In *Proceedings of the Workshop on Applications of Description Logics (ADL'04)*, Ulm, Germany, September 2004.

[27] Volker Haarslev, Ralf Möller, and Michael Wessel. Description Logic Inference Technology: Lessons Learned in the Trenches. In Horrocks et al. [40], pages 160–167.

[28] Christian Halashek-Wiener, Bijan Parsia, and Evren Sirin. Description Logics Reasoning with Syntactic Updates. In *Proc. of the 5th Int. Conf. on Ontologies, Databases, and Applications of Semantics (ODBASE 2006)*, Montpellier, France, October 2006. Sringer Verlag.

[29] Christian Halashek-Wiener, Bijan Parsia, and Evren Sirin. Towards Continuous Query Answering on the Semantic Web. In *Proc. of the 2nd Int. Workshop on Scalable Semantic Web Knowledge Base Systems (SSWS 2006)*, 2006.

[30] Jochen Heinsohn, Daniel Kudenko, Berhard Nebel, and Hans-Jürgen Profitlich. An Empirical Analysis of Terminological Representation Systems. Technical Report RR-92-16, German Research Center for Artificial Intelligence (DFKI), Kaiserslautern, Germany, 1992.

[31] I. Horrocks. Implementation and optimisation techniques. In Franz Baader, Diego Calvanese, Deborah McGuinness, Daniele Nardi, and Peter F. Patel-Schneider, editors, *The Description Logic Handbook: Theory, Implementation, and Applications*, pages 306–346. Cambridge University Press, 2003.

[32] Ian Horrocks. Using an Expressive Description Logic: FaCT or Fiction? In *Proc. of the Sixth Int. Conf. on Principles of Knowledge Representation and Reasoning (KR'98)*, pages 636–647, Trento, Italy, June 1998. Morgan Kaufmann Publishers, San Francisco, CA.

[33] Ian Horrocks. Applications of description logics: State of the art and research challenges. In Frithjof Dau, Marie-Laure Mugnier, and Gerd Stumme, editors, *Proc. of the 13th Int. Conf. on Conceptual Structures (ICCS'05)*, number 3596 in Lecture Notes in Artificial Intelligence, pages 78–90. Springer, 2005.

[34] Ian Horrocks, Oliver Kutz, and Ulrike Sattler. The Even More Irresistible $\mathcal{SROIQ}$. In *Proc. of the 10th Int. Conf. on Principles of Knowledge Representation and Reasoning (KR 2006)*, pages 57–57, Lake District, UK, June 2006.

[35] Ian Horrocks and Peter F. Patel-Schneider. Optimizing description logic subsumption. *J. of Logic and Computation*, 9(3):267–293, 1999.

[36] Ian Horrocks and Peter F. Patel-Schneider. Three theses of representation in the semantic web. In *Proc. of the Twelfth International World Wide Web Conference (WWW 2003)*, pages 39–47. ACM, 2003.

[37] Ian Horrocks, Peter F. Patel-Schneider, Harold Boley, Said Tabet, Benjamin Grosof, and Mike Dean. SWRL: A Semantic Web Rule Language Combining OWL and RuleML. W3C Member Submission, May 2004.

[38] Ian Horrocks, Peter F. Patel-Schneider, and Frank van Harmelen. From $\mathcal{SHIQ}$ and RDF to OWL: The making of a web ontology language. *Journal of Web Semantics*, 1(1):7–26, 2003.

[39] Ian Horrocks and Ulrike Sattler. A Tableaux Decision Procedure for $\mathcal{SHOIQ}$. In *Proc. of the 19th Int. Joint Conf. on Artificial Intelligence (IJCAI 2005)*, pages 448–453, 2005.

[40] Ian Horrocks, Ulrike Sattler, and Frank Wolter, editors. *Proc. of the Int. Workshop on Description Logics (DL'05)*, volume 147 of *CEUR*, Edinburgh, Scotland, July 2005.

[41] Ullrich Hustadt, Boris Motik, and Ulrike Sattler. Reasoning for Description

Logics around $\mathcal{SHIQ}$ in a Resolution Framework. Technical Report 3-8-04/04, Forschungszentrum Informatik (FZI), Karlsruhe, Germany, 2004.

[42] Aditya Kalyanpur, Bijan Parsia, Bernardo Cuenca-Grau, and Evren Sirin. Beyond Axioms: Fine-Grained Justifications for Arbitrary Entailments in OWL-DL. In Parsia et al. [53].

[43] Aditya Kalyanpur, Bijan Parsia, Evren Sirin, Bernardo Cuenca-Grau, and James Hendler. Swoop: A 'Web' Ontology Editing Browser. *Journal of Web Semantics*, 4(2):144–153, June 2006.

[44] Aditya Kalyanpur, Bijan Parsia, Evren Sirin, and James Hendler. Debugging unsatisfiable classes in owl ontologies. *Journal of Web Semantics*, 3(4):268–293, December 2005.

[45] Michael Kifer, Georg Lause, and James Wu. Logical Foundations of Object-Oriented and Frame-Based Languages. *Journal of the ACM*, 42(4):741–843, July 1995.

[46] A. Kiryakov, D. Ognyanov, and D. Manov. 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*, pages 182–192, New York City, USA, 2005. Springer.

[47] Thorsten Liebig. Reasoning Support for the Semantic Web. Technical Report ITR: I-FN-53, DoCoMo Euro-Labs GmbH, September 2004.

[48] Thorsten Liebig and Olaf Noppens. OntoTrack: A semantic approach for ontology authoring. *Journal of Web Semantics*, 3(2-3):116–131, October 2005.

[49] Thorsten Liebig, Friedrich von Henke, and Olaf Noppens. Explanation Support for OWL Authoring. In Explanation-Aware Computing: Papers from the 2005 Fall Symposium, ed. Thomas Roth-Berghofer and Stefan Schulz FS-05-04, American Association for Artificial Intelligence, Menlo Park, CA, USA, November 2005.

[50] Boris Motik and Ulrike Sattler. A Comparison of Techniques for Querying Large Description Logic ABoxes. In M. Hermann and A. Voronkov, editors, *Proc. of the 13th Int. Conf. on Logic Programming Artificial Intelligence and Reasoning (LPAR'06)*, LNCS, Phnom Penh, Cambodia, November 2006. Springer. To Appear.

[51] Boris Motik and Rudi Studer. KAON2 – A scalable reasoning tool for the Semantic Web. In *Proc. of the 2nd European Semantic Web Conference (ESWC'05)*, Heraklion, Greece, May 2005. poster session.

[52] Jeff Pan and Ian Horrocks. RDFS(FA) and RDF MT: Two semantics for RDFS. In Dieter Fensel, Katia Sycara, and John Mylopoulos, editors, *Proc. of the 2003 International Semantic Web Conference (ISWC 2003)*, number 2870 in Lecture Notes in Computer Science, pages 30–46. Springer, 2003.

[53] B. Parsia, U. Sattler, and D. Toman, editors. *Proc. of the Int. Workshop on Description Logics (DL'06)*, volume 189 of *CEUR*, Windermere, Lake District, UK, May 2006.

[54] Bijan Parsia, Evren Sirin, and Aditya Kalyanpur. Debugging OWL Ontologies. In *Proc. of the 14th Int. World Wide Web Conf. (WWW2005)*, Chiba, Japan, May 2005.

[55] Peter F. Patel-Schneider and Bill Swartout. Description Logic Specification from the KRSS Effort. Working version (draft), 1993.

[56] Alexandre Riazanov and Andrei Voronkov. The design and implementation of vampire. *AI Communications*, 15(2–3), 2002.

[57] E. Sirin, B. Cuenca Grau, and B. Parsia. From wine to water: Optimizing Description Logic reasoning for nominals. In *Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR'06)*, Lake District, UK, June 2006.

[58] Evren Sirin, Bijan Parsia, Bernardo Grau, Aditya Kalyanpur, and Y. Katz. Pellet: A practical OWL DL reasoner. *Journal of Web Semantics*, 2006. submitted.

[59] Dmitry Tsarkov. FaCT++ Software. `http://wonderweb.semanticweb.org/software.shtml`, 2003.

[60] Dmitry Tsarkov and Ian Horrocks. Reasoner Prototype – Implementing a new reasoner with datatypes support. WonderWeb Deliverable No. 13, Sept. 2003.

[61] Dmitry Tsarkov and Ian Horrocks. Efficient Reasoning with Range and Domain Constraints. In Haarselv and Möller [23], pages 41–50.

[62] Dmitry Tsarkov and Ian Horrocks. FaCT++ Description Logic Reasoner: System Description. In *Proc. of the Int. Joint Conference on Automated Reasoning (IJCAR'06)*, 2006. To Appear.

[63] Dmitry Tsarkov, Alexandre Riazanov, Sean Bechhofer, and Ian Horrocks. Using Vampire to Reason With OWL. In *Proc. of the 2004 International Semantic Web Conference (ISWC 2004)*, number 3298 in Lecture Notes in Computer Science, pages 471–485, Hiroshima, Japan, 2004. Springer Verlag.

[64] Anni-Yasmin Turhan and Christian Kissig. SONIC—System Description. In Haarselv and Möller [23].

[65] Taowei David Wang, Bijan Parsia, and James Hendler. A Survey of the Web Ontology Landscape. In *Proc. of 5th Int. Semantic Web Conference (ISWC'06)*, Athens, GA, USA, November 2006. Springer Verlag. to appear.

[66] Timo Weithöner. ABox Benchmarking OWL Reasoners. Technical Report ITR: I-FN-83, DoCoMo Euro-Labs GmbH, September 2006.

[67] Timo Weithöner, Thorsten Liebig, Marko Luther, and Sebastian Böhm. Whats Wrong with OWL Benchmarks? In *Proc. of the Workshop on Scalable Semantic Web Services (SSWS 2006)*, Athens, Georgia, USA, November 2006. to appear.

[68] M. Wessel and R. Möller. A Flexible DL-based Architecture for Deductive Information Systems. In G. Sutcliffe, R. Schmidt, and S. Schulz, editors, *Proc. of the*

IJCAR-06 Workshop on Empirically Successful Computerized Reasoning (ESCoR), pages 92–111, 2006.

[69] Michael Wessel and Ralf Möller. A high performance semantic web query answering engine. In Horrocks et al. [40].

[70] Evgeny Zolin. DL Complexity Navigator. School of Computer Science, University of Manchester, UK, 2006. `http://www.cs.man.ac.uk/~ezolin/logic/complexity.html`.

A. Hard TBox Tests

Remark: Within the abstract DL syntax usually the following, logical equivalent expressions are used in order to denote typicall OWL axioms (such as domain and range restrictions, property attributes, etc.):

$\top \sqsubseteq \forall \mathsf{r}.\mathrm{C}$ states that property r has range C
$\top \sqsubseteq \forall \mathsf{r}^{-1}.\mathrm{D}$ states that property r has domain D
$\top \sqsubseteq \leq 1\ \mathsf{r}.\top$ states that r is a functional property

Test Case 1a & 1b

Query 1a: F satisfiable
Query 1b: G unsatisfiable

$$
\begin{aligned}
\mathsf{rs2} &\sqsubseteq \mathsf{r} \\
\mathsf{rs3} &\sqsubseteq \mathsf{r} \\
\mathsf{rs4} &\sqsubseteq \mathsf{r} \\
\mathsf{rs1} &\sqsubseteq \mathsf{r} \\
\mathsf{rs5} &\sqsubseteq \mathsf{r} \\
\mathsf{rs6} &\equiv \mathsf{r}^{-1} \\
\top &\sqsubseteq \forall \mathsf{rs2}.\mathrm{D} \\
\top &\sqsubseteq \forall \mathsf{rs3}.\mathrm{H} \\
\top &\sqsubseteq \forall \mathsf{rs4}.(\mathrm{H} \sqcup \mathrm{E}) \\
\top &\sqsubseteq \forall \mathsf{rs1}.\mathrm{C} \\
\top &\sqsubseteq \forall \mathsf{rs5}.(\mathrm{E} \sqcap \neg \mathrm{C} \sqcap \mathrm{D}) \\
\mathrm{F} &\equiv \exists \mathsf{r}.\mathrm{A} \sqcap \geq 3\ \mathsf{rs1}.\top \sqcap \geq 3\ \mathsf{rs2}.\top \sqcap \geq 3\ \mathsf{rs3}.\top \sqcap = 4\ \mathsf{rs4}.\top \sqcap \geq 2\ \mathsf{rs5}.\top \sqcap \\
&\quad \exists \mathsf{r}.\mathrm{E} \sqcap \mathrm{C} \sqcap \mathrm{D} \sqcap \leq 4\ \mathsf{r}.\top \sqcap \forall \mathsf{r}.\geq 1\ \mathsf{r}.\top \sqcap \forall \mathsf{r}.\forall \mathsf{r}.\forall \mathsf{rs6}.\mathrm{C} \sqcup \mathrm{A} \\
\mathrm{G} &\equiv \exists \mathsf{r}.\mathrm{A} \sqcap \geq 3\ \mathsf{rs1}.\top \sqcap \geq 3\ \mathsf{rs2}.\top \sqcap \geq 3\ \mathsf{rs3}.\top \sqcap = 4\ \mathsf{rs4}.\top \sqcap \geq 3\ \mathsf{rs5}.\top \sqcap \\
&\quad \exists \mathsf{r}.\mathrm{E} \sqcap \mathrm{C} \sqcap \mathrm{D} \sqcap \leq 4\ \mathsf{r}.\top \sqcap \forall \mathsf{r}.\geq 1\ \mathsf{r}.\top \sqcap \forall \mathsf{r}.\forall \mathsf{r}.\forall \mathsf{rs6}.\mathrm{C} \sqcup \mathrm{A}
\end{aligned}
$$

Test Case 2a & 2b

Query 2a: V satisfiable
Query 2b: W unsatisfiable

$$
\begin{aligned}
\mathrm{P} &\equiv \geq 3\ \mathsf{q}.\top \\
\mathrm{A} &\equiv \geq 5\ \mathsf{q}.\top \\
\mathrm{Q} &\equiv \leq 2\ \mathsf{q}.\top \\
\mathrm{L} &\equiv \leq 1\ \mathsf{q}.\top \\
\mathrm{M} &\equiv \geq 2\ \mathsf{q}.\top \sqcap \leq 4\ \mathsf{q}.\top \\
\mathrm{I} &\equiv \mathrm{L} \sqcup \mathrm{M} \sqcup \mathrm{A} \\
\mathrm{F} &\equiv \leq 3\ \mathsf{q}.\top \\
\mathsf{rnotF} &\sqsubseteq \mathsf{r}
\end{aligned}
$$

$$
\begin{aligned}
\mathsf{rMorQP} &\sqsubseteq \mathsf{r}\\
\mathsf{rP} &\sqsubseteq \mathsf{r}\\
\mathsf{rF} &\sqsubseteq \mathsf{r}\\
\mathsf{rI} &\sqsubseteq \mathsf{r}\\
\mathsf{rMnotL} &\sqsubseteq \mathsf{r}\\
\mathsf{rMP} &\sqsubseteq \mathsf{r}\\
\top &\sqsubseteq \forall \mathsf{rnotF}.\neg \mathrm{F}\\
\top &\sqsubseteq \forall \mathsf{rMorQP}.(\mathrm{M} \sqcap (\mathrm{P} \sqcup \mathrm{Q}))\\
\top &\sqsubseteq \forall \mathsf{rP}.\mathrm{P}\\
\top &\sqsubseteq \forall \mathsf{rF}.\mathrm{F}\\
\top &\sqsubseteq \forall \mathsf{rI}.\mathrm{I}\\
\top &\sqsubseteq \forall \mathsf{rMnotL}.(\mathrm{M} \sqcap \neg \mathrm{L})\\
\top &\sqsubseteq \forall \mathsf{rMP}.(\mathrm{M} \sqcap \mathrm{P})\\
\mathrm{V} &\equiv \exists \mathsf{r}.\mathrm{L} \sqcap \geq 4\ \mathsf{rMnotL}.\top \sqcap \geq 3\ \mathsf{rF}.\top \sqcap \geq 3\ \mathsf{rP}.\top \sqcap = 4\ \mathsf{rMorQP}.\top\\
&\quad \sqcap \geq 2\ \mathsf{rnotF}.\top \sqcap \geq 4\ \mathsf{rI}.\top \sqcap \leq 2\ \mathsf{rMP}.\top \sqcap \leq 6\ \mathsf{r}.\top\\
\mathrm{W} &\equiv \exists \mathsf{r}.\mathrm{L} \sqcap \geq 4\ \mathsf{rMnotL}.\top \sqcap \geq 4\ \mathsf{rF}.\top \sqcap \geq 3\ \mathsf{rP}.\top \sqcap = 4\ \mathsf{rMorQP}.\top\\
&\quad \sqcap \geq 2\ \mathsf{rnotF}.\top \sqcap \geq 4\ \mathsf{rI}.\top \sqcap \leq 2\ \mathsf{rMP}.\top \sqcap \leq 5\ \mathsf{r}.\top
\end{aligned}
$$

Test Case 3a & 3b

Query 3a: C satisfiable
Query 3b: U unsatisfiable

$$
\begin{aligned}
\mathrm{P1} &\sqsubseteq \neg \mathrm{P2} \sqcup \mathrm{P3} \sqcup \mathrm{P4} \sqcup \mathrm{P5}\\
\mathrm{P2} &\sqsubseteq \neg \mathrm{P3} \sqcup \mathrm{P4} \sqcup \mathrm{P5}\\
\mathrm{P3} &\sqsubseteq \neg \mathrm{P4} \sqcup \mathrm{P5}\\
\mathrm{C} &\sqsubseteq \exists \mathsf{r}.\mathrm{P1} \sqcap \exists \mathsf{r}.\mathrm{P2} \sqcap \exists \mathsf{r}.\mathrm{P3} \sqcap \exists \mathsf{r}.\mathrm{P1} \sqcap \mathrm{P} \sqcap \exists \mathsf{r}.\mathrm{P2} \sqcap \mathrm{P} \sqcap\\
&\quad \exists \mathsf{r}.\mathrm{P3} \sqcap \mathrm{P} \sqcap \leq 3\ \mathsf{r}.\top\\
\mathrm{U} &\sqsubseteq \exists \mathsf{r}.\mathrm{P1} \sqcap \exists \mathsf{r}.\mathrm{P2} \sqcap \exists \mathsf{r}.\mathrm{P3} \sqcap \exists \mathsf{r}.\mathrm{P4} \sqcap\\
&\quad \exists \mathsf{r}.\mathrm{P1} \sqcap \mathrm{P} \sqcap \exists \mathsf{r}.\mathrm{P2} \sqcap \mathrm{P} \sqcap \exists \mathsf{r}.\mathrm{P3} \sqcap \mathrm{P} \sqcap \leq 3\ \mathsf{r}.\top
\end{aligned}
$$

Test Case 4

Query 4: L satisfiable

$$
\begin{aligned}
\mathsf{r} &\equiv \mathsf{s}^{-1}\\
\mathrm{K} &\equiv \mathrm{L} \sqcap \forall \mathsf{r}.\bot\\
\mathrm{L} &\equiv \leq 1\ \mathsf{r}.\top \sqcap \forall \mathsf{s}.\mathrm{L} \sqcap \geq 1\ \mathsf{s}.\top
\end{aligned}
$$

Test Case 5

Query 5: M2 unsatisfiable

$$
\begin{aligned}
\textsc{MyBottom} &\sqsubseteq \bot\\
\mathsf{qi} &\equiv \mathsf{q}^{-1}
\end{aligned}
$$

$$\begin{aligned} \text{M2} &\equiv \text{A} \sqcap \geq 1\ \mathsf{q}.\top \sqcap \forall \mathsf{q}.\text{M2} \sqcap \forall \mathsf{qi}.\textsc{NotM2} \\ \textsc{MyBottom} &\equiv \text{M2} \sqcap \textsc{NotM2} \end{aligned}$$

Test Case 6

Query 6: E unsatisfiable

$$\begin{aligned} \textsc{Al1r} &\equiv \geq 1\ \mathsf{r}.\top \\ \textsc{AllrinegA} &\equiv \forall \mathsf{ri}.\textsc{negA} \\ \text{A} &\equiv \geq 1\ \mathsf{q}.\top \\ \textsc{negA} &\equiv = 0\ \mathsf{q}.\top \\ \textsc{Allr} &\equiv \forall \mathsf{r}.\textsc{Allri1} \\ \textsc{Allri1} &\equiv \forall \mathsf{ri}.\textsc{AllrinegA} \\ \mathsf{ri} &\equiv \mathsf{r}^{-1} \\ \text{E} &\equiv \text{A} \sqcap \exists \mathsf{r}.\textsc{Al1r} \sqcap \forall \mathsf{r}.\textsc{Allr} \end{aligned}$$

Test Case 7

Query 7: N $\sqsubseteq$ NX

$$\begin{aligned} \text{N} &\sqsubseteq \forall \mathsf{r}.\{j\} \\ \text{NX} &\equiv \leq 1\ \mathsf{r}.\top \end{aligned}$$

Test Case 8

Query 8: I is instance of F: F(I)

$$\begin{aligned} &\textsc{negV}(T) \\ &\text{V}(O) \\ &\mathsf{h}(I, P) \\ &\mathsf{h}(I, O) \\ &\mathsf{h}(O, P) \\ &\mathsf{h}(P, T) \\ \textsc{ShnegV} &\equiv \exists \mathsf{h}.\textsc{negV} \\ \text{F} &\equiv \exists \mathsf{h}.\textsc{VandShnegV} \\ \text{V} &\equiv \geq 1\ \mathsf{r}.\top \\ \textsc{VandShnegV} &\equiv \text{V} \sqcap \textsc{ShnegV} \\ \textsc{negV} &\equiv = 0\ \mathsf{r}.\top \end{aligned}$$

Test Case 9

Query 9: Somer3CSomer4D $\sqsubseteq$ Allr5c

$$\begin{aligned} \mathsf{r5} &\sqsubseteq \mathsf{r2} \\ \mathsf{r3} &\sqsubseteq \mathsf{r1} \\ \mathsf{r4} &\sqsubseteq \mathsf{r2} \end{aligned}$$

$$
\begin{aligned}
\mathsf{r4} &\sqsubseteq \mathsf{r1} \\
\top &\sqsubseteq \leq 1\ \mathsf{r1}.\top \\
\top &\sqsubseteq \leq 1\ \mathsf{r2}.\top \\
\textsc{Somer3notCSomer4D} &\equiv \exists \mathsf{r3}.\textsc{NotC} \sqcap \exists \mathsf{r4}.\mathrm{D} \\
\textsc{NotC} &\equiv \geq 1\ \mathsf{p}.\top \\
\textsc{Allr5D} &\equiv \forall \mathsf{r5}.\mathrm{D} \\
\textsc{Somer3CSomer4D} &\equiv \exists \mathsf{r3}.\mathrm{C} \sqcap \exists \mathsf{r4}.\mathrm{D} \\
\mathrm{C} &\equiv = 0\ \mathsf{p}.\top \\
\textsc{Allr5C} &\equiv \forall \mathsf{r5}.\mathrm{C}
\end{aligned}
$$

Test Case 10a & 10b

Query 10a: X2 unsatisfiable
Query 10b: X3 satisfiable

$$
\begin{aligned}
\mathrm{C1} &\sqsubseteq \neg\mathrm{C2} \sqcap \neg\mathrm{C3} \\
\mathrm{C2} &\sqsubseteq \neg\mathrm{C3} \\
\mathrm{X2} &\equiv \exists \mathsf{r}.\mathrm{C1} \sqcap \exists \mathsf{r}.\mathrm{C2} \sqcap \exists \mathsf{r}.\mathrm{C3} \sqcap \exists \mathsf{r}.\mathrm{C4} \sqcap \exists \mathsf{r}.\mathrm{C5} \sqcap \exists \mathsf{r}.\mathrm{C6} \sqcap \exists \mathsf{r}.\mathrm{C7} \sqcap \exists \mathsf{r}.\mathrm{C8} \sqcap \\
&\quad \exists \mathsf{r}.\mathrm{C9} \sqcap \exists \mathsf{r}.\mathrm{C10} \sqcap \exists \mathsf{r}.\mathrm{C11} \sqcap \exists \mathsf{r}.\mathrm{C12} \sqcap \exists \mathsf{r}.\mathrm{C13} \sqcap \exists \mathsf{r}.\mathrm{C14} \sqcap \exists \mathsf{r}.\mathrm{C15} \sqcap \\
&\quad \exists \mathsf{r}.\mathrm{C16} \sqcap \exists \mathsf{r}.\mathrm{C17} \sqcap \exists \mathsf{r}.\mathrm{C18} \sqcap \leq 2\ \mathsf{r}.\top \\
\mathrm{X3} &\equiv \exists \mathsf{r}.\mathrm{C1} \sqcap \exists \mathsf{r}.\mathrm{C2} \sqcap \exists \mathsf{r}.\mathrm{C3} \sqcap \exists \mathsf{r}.\mathrm{C4} \sqcap \exists \mathsf{r}.\mathrm{C5} \sqcap \exists \mathsf{r}.\mathrm{C6} \sqcap \exists \mathsf{r}.\mathrm{C7} \sqcap \exists \mathsf{r}.\mathrm{C8} \sqcap \\
&\quad \exists \mathsf{r}.\mathrm{C9} \sqcap \exists \mathsf{r}.\mathrm{C10} \sqcap \exists \mathsf{r}.\mathrm{C11} \sqcap \exists \mathsf{r}.\mathrm{C12} \sqcap \exists \mathsf{r}.\mathrm{C13} \sqcap \exists \mathsf{r}.\mathrm{C14} \sqcap \exists \mathsf{r}.\mathrm{C15} \sqcap \\
&\quad \exists \mathsf{r}.\mathrm{C16} \sqcap \exists \mathsf{r}.\mathrm{C17} \sqcap \exists \mathsf{r}.\mathrm{C18} \sqcap \leq 3\ \mathsf{r}.\top
\end{aligned}
$$

Test Case 11

Query 11: $\textsc{EmptyI} \equiv \top$ / $\textsc{EmptyO} \equiv \bot$ / $\textsc{EmptyU} \equiv \bot$

$$
\begin{aligned}
\textsc{EmptyI} &\equiv (\sqcap) \quad \text{// empty intersection} \\
\textsc{EmptyO} &\equiv \{\} \quad \text{// empty enumeration} \\
\textsc{EmptyU} &\equiv (\sqcup) \quad \text{// empty union}
\end{aligned}
$$

Test Case 13

Query 13: $\textsc{SoccerDreamTeam} \sqsubseteq \textsc{TwoPlayerTeam}$ and
$\textsc{SoccerDreamTeam} \not\sqsubseteq \textsc{ThreePlayerTeam}$

$$
\begin{aligned}
\mathit{Beckenbauer} &\not\approx \mathit{Maradona} \\
\{\mathit{D10S}, \mathit{HandOfGod}, \mathit{Kaiser}\} &\equiv \{\mathit{Maradona}, \mathit{Beckenbauer}\} \\
\textsc{TwoPlayerTeam} &\equiv \geq 2\ \mathsf{player}.\top \\
\textsc{ThreePlayerTeam} &\equiv \geq 3\ \mathsf{player}.\top \\
\textsc{SoccerDreamTeam} &\equiv \exists \mathsf{player}.\{\mathit{D10S}\} \sqcap \exists \mathsf{player}.\{\mathit{HandOfGod}\} \sqcap \exists \mathsf{player}.\{\mathit{Kaiser}\}
\end{aligned}
$$

Test Case 14

Query 14: *Apple* is of type GreenColored: GreenColored(*Apple*)

$$
\begin{aligned}
&= 1\ \mathsf{hascolor}.\top(\mathit{Apple})\\
&\textsc{SomeColored}(\mathit{Apple})\\
&\mathit{Red} \not\approx \mathit{Green} \not\approx \mathit{Blue}\\
\textsc{SomeColored} &\sqsubseteq \forall \mathsf{hascolor}.\mathrm{RGB} \sqcap \mathrm{GB} \sqcap \mathrm{RG}\\
\textsc{GreenColored} &\equiv \exists \mathsf{hascolor}.\{\mathit{Green}\}\\
\mathrm{RGB} &\equiv \{\mathit{Red}, \mathit{Green}, \mathit{Blue}\}\\
\mathrm{GB} &\equiv \{\mathit{Green}, \mathit{Blue}\}\\
\mathrm{RG} &\equiv \{\mathit{Green}, \mathit{Red}\}
\end{aligned}
$$

Test Case 15

Query 15: Satisfiable is satisfiable
NotSatisfiable is unsatisfiable

$$
\begin{aligned}
&\textsc{Trans}(\mathsf{r})\\
\mathsf{invr} &\equiv \mathsf{r}^{-1}\\
\mathsf{invf} &\equiv \mathsf{f}^{-1}\\
\mathsf{f} &\sqsubseteq \mathsf{r}\\
\top &\sqsubseteq \leq 1\ \mathsf{invf}.\top\\
\top &\sqsubseteq \leq 1\ \mathsf{f}^{-1}.\top\\
\top &\sqsubseteq \leq 1\ \mathsf{f}.\top\\
\mathrm{D} &\equiv \mathrm{C} \sqcap \exists \mathsf{f}.\textsc{NotC}\\
\textsc{NotA} &\equiv \geq 1\ \mathsf{w}.\top\\
\mathrm{C} &\equiv = 0\ \mathsf{q}.\top\\
\mathrm{A} &\equiv = 0\ \mathsf{w}.\top\\
\textsc{MyBottom} &\equiv \bot\\
\textsc{SomeInvfD} &\equiv \exists \mathsf{invf}.\mathrm{D}\\
\textsc{NotC} &\equiv \geq 1\ \mathsf{q}.\top\\
\textsc{SomeInvfA} &\equiv \exists \mathsf{invf}.\mathrm{A}\\
\textsc{Satisfiable} &\equiv \textsc{NotA} \sqcap \exists \mathsf{invf}.\mathrm{A} \sqcap \forall \mathsf{invr}.\textsc{SomeInvfA}\\
\textsc{NotSatisfiable} &\equiv \textsc{NotC} \sqcap \exists \mathsf{invf}.\mathrm{D} \sqcap \forall \mathsf{invr}.\textsc{SomeInvfD}
\end{aligned}
$$

Test Case 16

Query 16: DullColored $\sqsubseteq$ NoOrLittleColored

$$
\begin{aligned}
\textsc{DullColored} &\sqsubseteq \forall \mathsf{hascolor}.\textsc{SingleColor}\\
\textsc{NoOrLittleColored} &\equiv \leq 1\ \mathsf{hascolor}.\top\\
\textsc{SingleColor} &\equiv \{\mathit{Red}\}
\end{aligned}
$$

Test Case 17

Query 17: $\top \equiv \bot$ (incoherent ontology)

$$\begin{aligned} \textsc{MyBottom} &\equiv \bot \\ \textsc{MyBottom} &\equiv \textsc{MyTop} \\ \textsc{MyTop} &\equiv \top \end{aligned}$$

Test Case 18

Query 18: $\textsc{AnotA} \equiv \top$

$$\textsc{AnotA} \equiv \mathrm{A} \sqcup \neg \mathrm{A}$$

Test Case 19

Query 19: Should signal an error since complex properties are not allowed to be used within at-most cardinality restrictions (A) and a transitive property can not be a sub-property of a functional property (OWL DL & OWL Lite)

$$\begin{aligned} \top &\sqsubseteq \leq 1\ \mathsf{q}.\top \\ \textsc{Trans}(&\mathsf{q\text{-}plus}) \\ \textsc{Trans}(&\mathsf{p\text{-}plus}) \\ \mathsf{q\text{-}plus} &\sqsubseteq \mathsf{q} \\ \mathsf{p\text{-}plus} &\sqsubseteq \mathsf{p} \\ \mathrm{A} &\equiv \leq 1\ \mathsf{p}.\top \end{aligned}$$

Test Case 20

Query 20: $\mathrm{K} \sqsubseteq \mathrm{L}$

$$\begin{aligned} \mathsf{r} &\equiv \mathsf{s}^{-1} \\ \mathrm{K} &\equiv \mathrm{L} \sqcap \forall \mathsf{r}.\bot \\ \mathrm{L} &\equiv \leq 1\ \mathsf{r}.\top \sqcap \forall \mathsf{s}.\mathrm{L} \sqcap \geq 1\ \mathsf{s}.\top \end{aligned}$$

Test Case 21

Query 21: *i1* is an instance of B but not of C: *i1*(B)

$$\begin{aligned} &\mathrm{A}(\mathit{i1}) \\ &\mathsf{r}(\mathit{i1}, "1") \\ &\mathsf{p}(\mathit{i1}, "2") \\ \top &\sqsubseteq \forall \mathsf{r}.\mathsf{xsd{:}integer} \\ \top &\sqsubseteq \forall \mathsf{p}.\mathsf{xsd{:}integer} \\ \top &\sqsubseteq \leq 1 \mathsf{r}.\top \\ \mathsf{r} &\sqsubseteq \mathsf{s} \end{aligned}$$

$$\begin{aligned} \mathsf{p} &\sqsubseteq \mathsf{s} \\ \mathrm{C} &\equiv \geq 3\ \mathsf{s}.\top \\ \mathrm{B} &\equiv \geq 2\ \mathsf{s}.\top \end{aligned}$$

Test Case 22

Query 22: $\mathrm{A} \equiv \bot$

$$\begin{aligned} \top &\sqsubseteq \forall \mathsf{age.xsd{:}integer} \\ \mathrm{A} &\equiv \forall \mathsf{age.xsd{:}nonNegativeInteger} \sqcap \exists \mathsf{age.xsd{:}negativeInteger} \end{aligned}$$

Test Case 23

Query 23: $\textsc{Equiv}\mathrm{A} \equiv \mathrm{A}$

$$\begin{aligned} \textsc{Equiv}\mathrm{A} &\equiv \neg \mathrm{B} \\ \neg \mathrm{A} \sqcap \neg \mathrm{B} &\equiv \bot \\ \mathrm{A} &\equiv \mathrm{C} \sqcup \mathrm{D} \\ \bot &\sqsubseteq \mathrm{B} \sqcap \mathrm{A} \end{aligned}$$

Test Case 25

Query 25: a is of type F: $a(\mathrm{F})$

$$\begin{aligned} \mathit{i1} &\not\approx \mathit{i2} \\ \top &\sqsubseteq \forall \mathsf{q}.\mathrm{A} \\ \mathsf{p} &\sqsubseteq \mathsf{q} \\ \mathsf{n} &\sqsubseteq \mathsf{q} \\ \mathrm{F} &\equiv \exists \mathsf{r}.(\exists \mathsf{n}.\{\mathit{i1}\} \sqcap \exists \mathsf{p}.\{\mathit{i2}\}) \\ \mathrm{G} &\equiv \exists \mathsf{r}.(\geq 2\ \mathsf{q}.\top) \\ \mathrm{A} &\equiv \{\mathit{i1}, \mathit{i2}\} \\ \mathsf{r}(a, c) & \\ \mathsf{r}(a, b) & \\ \mathsf{n}(c, \mathit{i3}) & \\ \mathsf{p}(c, \mathit{i2}) & \\ \mathsf{p}(b, \mathit{i3}) & \\ \mathsf{n}(b, \mathit{i1}) & \end{aligned}$$

Test Case 26

Query 26: $\mathrm{F} \sqsubseteq \mathrm{G2}$

$$\begin{aligned} \mathrm{F} &\equiv \exists \mathsf{r}.\{\mathit{i1}\} \sqcap \exists \mathsf{r}.\{\mathit{i2}\} \sqcap \exists \mathsf{r}.\{\mathit{i3}\} \sqcap \exists \mathsf{r}.\{\mathit{i4}\} \\ \mathrm{G3} &\equiv \geq 3\ \mathsf{r}.\top \\ \mathrm{A} &\equiv \{\mathit{i1}, \mathit{i2}, \mathit{i3}, \mathit{i4}\} \\ \mathrm{G4} &\equiv \geq 4\ \mathsf{r}.\top \end{aligned}$$

$$
\begin{aligned}
\mathrm{B} &\equiv \{e1, e2, e3, e4, e5, e6, e7\} \\
\mathrm{G2} &\equiv \geq 2\ \mathsf{r}.\top \\
e2 &\not\approx e5 \\
e3 &\not\approx e7 \\
\bot &\sqsubseteq \mathrm{B} \sqcap \neg \mathrm{A}
\end{aligned}
$$

Test Case 27

Query 27: $\mathsf{q} \sqsubseteq \mathsf{p}$ or alternatively $\mathrm{T2} \sqsubseteq \mathrm{T1}$

$$
\begin{aligned}
\top &\sqsubseteq \forall \mathsf{p}.\{b, c\} \\
\top &\sqsubseteq \forall \mathsf{q}.\{b\} \\
\top &\sqsubseteq \forall \mathsf{p}^{-1}.\{a\} \\
\top &\sqsubseteq \forall \mathsf{q}^{-1}.\{a\} \\
\mathrm{T1} &\equiv \geq 1\ \mathsf{p}.\top \\
\mathrm{T2} &\equiv \geq 1\ \mathsf{q}.\top \\
&\mathsf{p}(a, b) \\
&\mathsf{p}(a, c) \\
&\mathsf{q}(a, b)
\end{aligned}
$$

Test Case 28

Query 28: A is satisfiable

$$
\begin{aligned}
\mathrm{A} \equiv\ & \exists \mathsf{p}0.\forall \mathsf{p}1.\forall \mathsf{p}2.\forall \mathsf{p}3.\forall \mathsf{p}4.\forall \mathsf{p}5.\forall \mathsf{p}6.\mathrm{C}01 \sqcap \\
& \exists \mathsf{p}0.\forall \mathsf{p}1.\forall \mathsf{p}2.\forall \mathsf{p}3.\forall \mathsf{p}4.\forall \mathsf{p}5.\forall \mathsf{p}6.\mathrm{C}02 \sqcap \\
& \exists \mathsf{p}0.\forall \mathsf{p}1.\forall \mathsf{p}2.\forall \mathsf{p}3.\forall \mathsf{p}4.\forall \mathsf{p}5.\forall \mathsf{p}6.\mathrm{C}03 \sqcap \\
& \exists \mathsf{p}0.\forall \mathsf{p}1.\forall \mathsf{p}2.\forall \mathsf{p}3.\forall \mathsf{p}4.\forall \mathsf{p}5.\forall \mathsf{p}6.\mathrm{C}04 \sqcap \\
& \exists \mathsf{p}0.\forall \mathsf{p}1.\forall \mathsf{p}2.\forall \mathsf{p}3.\forall \mathsf{p}4.\forall \mathsf{p}5.\forall \mathsf{p}6.\mathrm{C}05 \sqcap \\
& \exists \mathsf{p}0.\forall \mathsf{p}1.\forall \mathsf{p}2.\forall \mathsf{p}3.\forall \mathsf{p}4.\forall \mathsf{p}5.\forall \mathsf{p}6.\mathrm{C}06 \sqcap \\
& \exists \mathsf{p}0.\forall \mathsf{p}1.\forall \mathsf{p}2.\forall \mathsf{p}3.\forall \mathsf{p}4.\forall \mathsf{p}5.\forall \mathsf{p}6.\mathrm{C}07 \sqcap \\
& \exists \mathsf{p}0.\forall \mathsf{p}1.\forall \mathsf{p}2.\forall \mathsf{p}3.\forall \mathsf{p}4.\forall \mathsf{p}5.\forall \mathsf{p}6.\mathrm{C}08 \sqcap \\
& \forall \mathsf{p}0.(\exists \mathsf{p}1.\forall \mathsf{p}2.\forall \mathsf{p}3.\forall \mathsf{p}4.\forall \mathsf{p}5.\forall \mathsf{p}6.\mathrm{C}11 \sqcap \\
& \quad \exists \mathsf{p}1.\forall \mathsf{p}2.\forall \mathsf{p}3.\forall \mathsf{p}4.\forall \mathsf{p}5.\forall \mathsf{p}6.\mathrm{C}12 \sqcap \\
& \quad \exists \mathsf{p}1.\forall \mathsf{p}2.\forall \mathsf{p}3.\forall \mathsf{p}4.\forall \mathsf{p}5.\forall \mathsf{p}6.\mathrm{C}13 \sqcap \\
& \quad \exists \mathsf{p}1.\forall \mathsf{p}2.\forall \mathsf{p}3.\forall \mathsf{p}4.\forall \mathsf{p}5.\forall \mathsf{p}6.\mathrm{C}14 \sqcap \\
& \quad \exists \mathsf{p}1.\forall \mathsf{p}2.\forall \mathsf{p}3.\forall \mathsf{p}4.\forall \mathsf{p}5.\forall \mathsf{p}6.\mathrm{C}15 \sqcap \\
& \quad \forall \mathsf{p}1.(\exists \mathsf{p}2.\forall \mathsf{p}3.\forall \mathsf{p}4.\forall \mathsf{p}5.\forall \mathsf{p}6.\mathrm{C}21 \sqcap \\
& \quad\quad \exists \mathsf{p}2.\forall \mathsf{p}3.\forall \mathsf{p}4.\forall \mathsf{p}5.\forall \mathsf{p}6.\mathrm{C}22 \sqcap \\
& \quad\quad \exists \mathsf{p}2.\forall \mathsf{p}3.\forall \mathsf{p}4.\forall \mathsf{p}5.\forall \mathsf{p}6.\mathrm{C}23 \sqcap \\
& \quad\quad \exists \mathsf{p}2.\forall \mathsf{p}3.\forall \mathsf{p}4.\forall \mathsf{p}5.\forall \mathsf{p}6.\mathrm{C}24 \sqcap \\
& \quad\quad \exists \mathsf{p}2.\forall \mathsf{p}3.\forall \mathsf{p}4.\forall \mathsf{p}5.\forall \mathsf{p}6.\mathrm{C}25 \sqcap \\
& \quad\quad \forall \mathsf{p}2.(\exists \mathsf{p}3.\forall \mathsf{p}4.\forall \mathsf{p}5.\forall \mathsf{p}6.\mathrm{C}31 \sqcap \\
& \quad\quad\quad \exists \mathsf{p}3.\forall \mathsf{p}4.\forall \mathsf{p}5.\forall \mathsf{p}6.\mathrm{C}32 \sqcap \\
& \quad\quad\quad \exists \mathsf{p}3.\forall \mathsf{p}4.\forall \mathsf{p}5.\forall \mathsf{p}6.\mathrm{C}33 \sqcap
\end{aligned}
$$

$$
\begin{aligned}
\forall \mathsf{p3}.(&\exists \mathsf{p4}.\forall \mathsf{p5}.\forall \mathsf{p6}.\mathrm{C41} \sqcap \\
&\exists \mathsf{p4}.\forall \mathsf{p5}.\forall \mathsf{p6}.\mathrm{C42} \sqcap \\
&\exists \mathsf{p4}.\forall \mathsf{p5}.\forall \mathsf{p6}.\mathrm{C43} \sqcap \\
&\forall \mathsf{p4}.(\exists \mathsf{p5}.\forall \mathsf{p6}.\mathrm{C51} \sqcap \\
&\quad \exists \mathsf{p5}.\forall \mathsf{p6}.\mathrm{C52} \sqcap \\
&\quad \exists \mathsf{p5}.\forall \mathsf{p6}.\mathrm{C53} \sqcap \\
&\quad \forall \mathsf{p5}.(\exists \mathsf{p6}.\mathrm{C61} \sqcap \\
&\quad\quad \exists \mathsf{p6}.\mathrm{C62} \sqcap \\
&\quad\quad \exists \mathsf{p6}.\mathrm{C63} \sqcap \\
&\quad\quad \exists \mathsf{p6}.\mathrm{C64} \sqcap \\
&\quad\quad \exists \mathsf{p6}.\mathrm{C65})))))
\end{aligned}
$$

Test Case 29a & 29b

Query 29a: F-LITE is satisfiable
Query 29b: F-LITE2 is unsatisfiable

$$
\begin{aligned}
\mathsf{rs5} &\equiv \mathsf{r}^{-1} \\
\mathsf{rs4} &\sqsubseteq \mathsf{r} \\
\mathsf{rs2} &\sqsubseteq \mathsf{r} \\
\mathsf{rs1} &\sqsubseteq \mathsf{r} \\
\mathsf{rs3} &\sqsubseteq \mathsf{r} \\
\top &\sqsubseteq \forall \mathsf{rs4}.\mathrm{DRS4} \\
\top &\sqsubseteq \forall \mathsf{rs2}.\mathrm{D} \\
\top &\sqsubseteq \forall \mathsf{rs1}.\mathrm{C} \\
\top &\sqsubseteq \forall \mathsf{rs3}.\mathrm{DRS5} \\
\top &\sqsubseteq \forall \mathsf{rs3}.\mathrm{H} \\
\mathrm{DRS4} &\equiv \neg(\neg \mathrm{H} \sqcap \neg \mathrm{E}) \\
\mathrm{ALLRS6} &\equiv \neg(\neg \mathrm{C} \sqcap \neg \mathrm{D}) \\
\mathrm{DRS5} &\equiv \mathrm{E} \sqcap \neg(\mathrm{C} \sqcup \mathrm{D}) \\
\mathrm{EANDC} &\equiv \mathrm{E} \sqcap \mathrm{C} \\
\mathrm{ALLR} &\equiv \forall \mathsf{r}.\mathrm{ALLRS6} \\
\mathrm{ATL1} &\equiv \geq 1\ \mathsf{r}.\top \\
\text{F-LITE} &\equiv \exists \mathsf{r}.\mathrm{A} \sqcap \leq 1\ \mathsf{rs1}.\top \sqcap \geq 1\ \mathsf{rs2}.\top \sqcap \geq 1\ \mathsf{rs3}.\top \sqcap = 1\ \mathsf{rs4}.\top \sqcap \\
&\quad \geq 1\ \mathsf{rs5}.\top \sqcap \exists \mathsf{r}.\mathrm{EANDC} \sqcap \forall \mathsf{r}.\mathrm{ATL1} \sqcap \forall \mathsf{r}.\mathrm{ALLR} \\
\text{F-LITE2} &\equiv \exists \mathsf{r}.\mathrm{A} \sqcap \leq 1\ \mathsf{rs1}.\top \sqcap \geq 1\ \mathsf{rs2}.\top \sqcap \geq 1\ \mathsf{rs3}.\top \sqcap = 1\ \mathsf{rs4}.\top \sqcap \\
&\quad \geq 1\ \mathsf{rs5}.\top \sqcap \exists \mathsf{r}.\mathrm{EANDC} \sqcap \forall \mathsf{r}.\mathrm{ATL1} \sqcap \forall \mathsf{r}.\mathrm{ALLR} \sqcap \leq 1\ \mathsf{r}.\top
\end{aligned}
$$

Test Case 30

Query 30:

$$
\begin{aligned}
&\mathrm{TRANS}(\mathsf{rest\text{-}plus}) \\
&\mathrm{TRANS}(\mathsf{inv\text{-}rest\text{-}plus}) \\
&\qquad\qquad \mathsf{rest} \sqsubseteq \mathsf{rest\text{-}plus}
\end{aligned}
$$

$$
\begin{aligned}
\mathsf{inv\text{-}first} &\equiv \mathsf{first}^{-1} \\
\mathsf{inv\text{-}rest} &\equiv \mathsf{rest}^{-1} \\
\mathsf{inv\text{-}rest\text{-}plus} &\equiv \mathsf{rest\text{-}plus}^{-1} \\
\top &\sqsubseteq \forall \mathsf{rest}^{-1}.\textsc{List} \\
\top &\sqsubseteq \forall \mathsf{first}^{-1}.\textsc{List} \\
\top &\sqsubseteq \leq 1\ \mathsf{rest}.\top \\
\top &\sqsubseteq \leq 1\ \mathsf{first}.\top \\
\textsc{NotBeginOfList} &\equiv \textsc{NonEmptyList} \sqcap \exists \mathsf{inv\text{-}rest}.\textsc{NonEmptyList} \\
\textsc{BeginOfList} &\equiv \textsc{NonEmptyList} \sqcap \forall \mathsf{inv\text{-}rest}.\textsc{NonList} \\
\textsc{BalancedTreeNode2D} &\equiv (\textsc{List} \sqcap \exists \mathsf{first}.\textsc{BalancedTreeNode2D} \sqcap \\
&\quad \exists \mathsf{rest}.\textsc{BalancedTreeNode2D}) \sqcup (\exists \mathsf{first}.\textsc{Leaf} \sqcap \\
&\quad \exists \mathsf{inv\text{-}first}.\exists \mathsf{rest}.\textsc{TreeLeafList}) \sqcup (\exists \mathsf{first}.\textsc{Leaf} \sqcap \\
&\quad \exists \mathsf{inv\text{-}rest}.\exists \mathsf{first}.\textsc{TreeLeafList}) \\
\neg\textsc{List} \sqcap \neg\textsc{NonList} &\equiv \bot \\
\textsc{DottedPair} &\equiv \textsc{List} \sqcap \exists \mathsf{first}.\textsc{NonList} \sqcap \exists \mathsf{rest}.\textsc{NonList} \\
\textsc{EndOfList} &\equiv \textsc{NonEmptyList} \sqcap \forall \mathsf{rest}.\textsc{EmptyList} \\
\textsc{List} &\equiv \textsc{EmptyList} \sqcup \textsc{NonEmptyList} \\
\textsc{MadonnaList} &\equiv \textsc{List} \sqcap \exists \mathsf{first}.\{\mathit{Madonna}\} \sqcup \exists \mathsf{rest\text{-}plus}.\textsc{MadonnaList} \\
\textsc{ActorsList} &\equiv \textsc{List} \sqcap \forall \mathsf{first}.\textsc{Actor} \sqcap \forall \mathsf{rest}.(\textsc{ActorsList} \sqcup \textsc{EmptyList}) \\
\textsc{NonEmptyList} &\equiv \geq 1\ \mathsf{first}.\top \sqcap \geq 1\ \mathsf{rest}.\top \\
\textsc{TreeLeafList} &\equiv \textsc{List} \sqcap \exists \mathsf{first}.\textsc{Leaf} \sqcap \exists \mathsf{rest}.\{\mathit{ListNil}\} \\
\textsc{MinLenght2List} &\equiv \textsc{List} \sqcap \exists \mathsf{rest}.\ \geq 1\ \mathsf{rest}.\top \\
\textsc{EmptyList} &\equiv \{\mathit{ListNil}\} \\
\textsc{SingersList} &\equiv \textsc{List} \sqcap \forall \mathsf{first}.\textsc{Singer} \sqcap \forall \mathsf{rest}.(\textsc{SingersList} \sqcup \textsc{EmptyList}) \\
\textsc{CyclicList} &\equiv \textsc{List} \sqcap \forall \mathsf{rest\text{-}plus}.\textsc{List} \\
\textsc{NonFlatList} &\equiv \textsc{List} \sqcap \exists \mathsf{first}.\textsc{NonEmptyList} \\
\bot &\sqsubseteq \textsc{Singer} \sqcap \textsc{EmptyList} \\
\bot &\sqsubseteq \textsc{Element} \sqcap \textsc{EmptyList} \\
\bot &\sqsubseteq \textsc{EmptyList} \sqcap \textsc{NonEmptyList} \\
\bot &\sqsubseteq \textsc{NonList} \sqcap \textsc{List}
\end{aligned}
$$

$\mathsf{rest}(\mathit{LeftLeftBranch}, \mathit{ListNil})$	$\mathsf{first}(\mathit{LeftLeftBranch}, \mathit{Eins})$
$\mathsf{rest}(\mathit{Act2}, \mathit{Act3})$	$\mathsf{first}(\mathit{Act2}, \mathit{ArnoldSchwarzenegger})$
$\mathsf{rest}(\mathit{DotList}, \mathit{AntiList1})$	$\mathsf{first}(\mathit{DotList}, \mathit{AntiList1})$
$\mathsf{rest}(\mathit{Sing1}, \mathit{Sing2})$	$\mathsf{first}(\mathit{Sing1}, \mathit{JanetJackson})$
$\mathsf{rest}(\mathit{Act3}, \mathit{Act4})$	$\mathsf{first}(\mathit{Act3}, \mathit{TomCruise})$
$\mathsf{rest}(\mathit{Sing2}, \mathit{Sing3})$	$\mathsf{first}(\mathit{Sing2}, \mathit{MikeOldfield})$
$\mathsf{rest}(\mathit{Act1}, \mathit{Act2})$	$\mathsf{first}(\mathit{Act1}, \mathit{SharonStone})$
$\mathsf{rest}(\mathit{Sing3}, \mathit{Sing4})$	$\mathsf{first}(\mathit{Sing3}, \mathit{Madonna})$
$\mathsf{rest}(\mathit{Sing4}, \mathit{ListNil})$	$\mathsf{first}(\mathit{Sing4}, \mathit{GloriaEstefan})$
$\mathsf{rest}(\mathit{TreeRoot}, \mathit{RightBranch})$	$\mathsf{first}(\mathit{TreeRoot}, \mathit{LeftBranch})$
$\mathsf{rest}(\mathit{Cycle2}, \mathit{Cycle3})$	$\mathsf{first}(\mathit{Cycle2}, \mathit{Elem2})$
$\mathsf{rest}(\mathit{LeftRightBranch}, \mathit{ListNil})$	$\mathsf{first}(\mathit{LeftRightBranch}, \mathit{Zwei})$

rest(*genid58*, *genid59*)
rest(*genid59*, *ListNil*)
rest(*Cycle3*, *Cycle1*)
rest(*Cycle1*, *Cycle2*)
rest(*Ano1*, *genid58*)
rest(*LeftBranch*, *LeftRightBranch*)
rest(*RightBranch*, *ListNil*)
rest(*Act4*, *Act5*)
rest(*Act5*, *ListNil*)
List(*LeftLeftBranch*)
List(*DotList*)
Element(*Elem1*)
ActorsList(*Act3*)
Actor(*Madonna*)
ActorsList(*Act1*)
Singer(*GloriaEstefan*)
NonList(*AntiList1*)
List(*Cycle2*)
List(*LeftRightBranch*)
Leaf(*Eins*)
List(*genid59*)
List(*Cycle3*)
Element(*Elem3*)
List(*Ano1*)
Actor(*SharonStone*)
List(*RightBranch*)
Actor(*ArnoldSchwarzenegger*)
Leaf(*Zwei*)
first(*genid58*, *Indiv2*)
first(*genid59*, *Indiv3*)
first(*Cycle3*, *Elem3*)
first(*Cycle1*, *Elem1*)
first(*Ano1*, *Indiv1*)
first(*LeftBranch*, *LeftLeftBranch*)
first(*RightBranch*, *Drei*)
first(*Act4*, *HenryFonda*)
first(*Act5*, *Madonna*)
ActorsList(*Act2*)
SingersList(*Sing1*)
NonList(*AntiList2*)
SingersList(*Sing2*)
Singer(*Madonna*)
SingersList(*Sing3*)
SingersList(*Sing4*)
List(*TreeRoot*)
Actor(*HenryFonda*)
Singer(*JanetJackson*)
List(*genid58*)
Actor(*TomCruise*)
Element(*Elem2*)
List(*Cycle1*)
Singer(*MikeOldfield*)
List(*LeftBranch*)
ActorsList(*Act4*)
Leaf(*Drei*)
ActorsList(*Act5*)

Liste der bisher erschienenen Ulmer Informatik-Berichte

Einige davon sind per FTP von `ftp.informatik.uni-ulm.de` erhältlich
Die mit * markierten Berichte sind vergriffen

List of technical reports published by the University of Ulm

Some of them are available by FTP from `ftp.informatik.uni-ulm.de`
Reports marked with * are out of print

91-01 *Ker-I Ko, P. Orponen, U. Schöning, O. Watanabe*
Instance Complexity

91-02* *K. Gladitz, H. Fassbender, H. Vogler*
Compiler-Based Implementation of Syntax-Directed Functional Programming

91-03* *Alfons Geser*
Relative Termination

91-04* *J. Köbler, U. Schöning, J. Toran*
Graph Isomorphism is low for PP

91-05 *Johannes Köbler, Thomas Thierauf*
Complexity Restricted Advice Functions

91-06* *Uwe Schöning*
Recent Highlights in Structural Complexity Theory

91-07* *F. Green, J. Köbler, J. Toran*
The Power of Middle Bit

91-08* *V.Arvind, Y. Han, L. Hamachandra, J. Köbler, A. Lozano, M. Mundhenk, A. Ogiwara, U. Schöning, R. Silvestri, T. Thierauf*
Reductions for Sets of Low Information Content

92-01* *Vikraman Arvind, Johannes Köbler, Martin Mundhenk*
On Bounded Truth-Table and Conjunctive Reductions to Sparse and Tally Sets

92-02* *Thomas Noll, Heiko Vogler*
Top-down Parsing with Simulataneous Evaluation of Noncircular Attribute Grammars

92-03 *Fakultät für Informatik*
17. Workshop über Komplexitätstheorie, effiziente Algorithmen und Datenstrukturen

92-04* *V. Arvind, J. Köbler, M. Mundhenk*
Lowness and the Complexity of Sparse and Tally Descriptions

92-05* *Johannes Köbler*
Locating P/poly Optimally in the Extended Low Hierarchy

92-06* *Armin Kühnemann, Heiko Vogler*
Synthesized and inherited functions -a new computational model for syntax-directed semantics

92-07* *Heinz Fassbender, Heiko Vogler*
A Universal Unification Algorithm Based on Unification-Driven Leftmost Outermost Narrowing

92-08* *Uwe Schöning*
On Random Reductions from Sparse Sets to Tally Sets

92-09* *Hermann von Hasseln, Laura Martignon*
Consistency in Stochastic Network

92-10 *Michael Schmitt*
A Slightly Improved Upper Bound on the Size of Weights Sufficient to Represent Any Linearly Separable Boolean Function

92-11 *Johannes Köbler, Seinosuke Toda*
On the Power of Generalized MOD-Classes

92-12 *V. Arvind, J. Köbler, M. Mundhenk*
Reliable Reductions, High Sets and Low Sets

92-13 *Alfons Geser*
On a monotonic semantic path ordering

92-14* *Joost Engelfriet, Heiko Vogler*
The Translation Power of Top-Down Tree-To-Graph Transducers

93-01 *Alfred Lupper, Konrad Froitzheim*
AppleTalk Link Access Protocol basierend auf dem Abstract Personal Communications Manager

93-02 *M.H. Scholl, C. Laasch, C. Rich, H.-J. Schek, M. Tresch*
The COCOON Object Model

93-03 *Thomas Thierauf, Seinosuke Toda, Osamu Watanabe*
On Sets Bounded Truth-Table Reducible to P-selective Sets

93-04 *Jin-Yi Cai, Frederic Green, Thomas Thierauf*
On the Correlation of Symmetric Functions

93-05 *K.Kuhn, M.Reichert, M. Nathe, T. Beuter, C. Heinlein, P. Dadam*
A Conceptual Approach to an Open Hospital Information System

93-06 *Klaus Ganer*
Rechnerunterstttzung fr die konzeptuelle Modellierung

93-07 *Ullrich Keler, Peter Dadam*
Towards Customizable, Flexible Storage Structures for Complex Objects

94-01 *Michael Schmitt*
On the Complexity of Consistency Problems for Neurons with Binary Weights

94-02 *Armin Khnemann, Heiko Vogler*
A Pumping Lemma for Output Languages of Attributed Tree Transducers

94-03 *Harry Buhrman, Jim Kadin, Thomas Thierauf*
On Functions Computable with Nonadaptive Queries to NP

94-04 *Heinz Faßbender, Heiko Vogler, Andrea Wedel*
Implementation of a Deterministic Partial E-Unification Algorithm for Macro Tree Transducers

94-05 *V. Arvind, J. Köbler, R. Schuler*
On Helping and Interactive Proof Systems

94-06 *Christian Kalus, Peter Dadam*
Incorporating record subtyping into a relational data model

94-07 *Markus Tresch, Marc H. Scholl*
A Classification of Multi-Database Languages

94-08 *Friedrich von Henke, Harald Rueß*
Arbeitstreffen Typtheorie: Zusammenfassung der Beiträge

94-09 *F.W. von Henke, A. Dold, H. Rueß, D. Schwier, M. Strecker*
Construction and Deduction Methods for the Formal Development of Software

94-10 *Axel Dold*
Formalisierung schematischer Algorithmen

94-11 *Johannes Köbler, Osamu Watanabe*
New Collapse Consequences of NP Having Small Circuits

94-12 *Rainer Schuler*
On Average Polynomial Time

94-13 *Rainer Schuler, Osamu Watanabe*
Towards Average-Case Complexity Analysis of NP Optimization Problems

94-14 *Wolfram Schulte, Ton Vullinghs*
Linking Reactive Software to the X-Window System

94-15 *Alfred Lupper*
Namensverwaltung und Adressierung in Distributed Shared Memory-Systemen

94-16 *Robert Regn*
Verteilte Unix-Betriebssysteme

94-17 *Helmuth Partsch*
Again on Recognition and Parsing of Context-Free Grammars:
Two Exercises in Transformational Programming

94-18 *Helmuth Partsch*
Transformational Development of Data-Parallel Algorithms: an Example

95-01 *Oleg Verbitsky*
On the Largest Common Subgraph Problem

95-02 *Uwe Schöning*
Complexity of Presburger Arithmetic with Fixed Quantifier Dimension

95-03 *Harry Buhrman,Thomas Thierauf*
The Complexity of Generating and Checking Proofs of Membership

95-04 *Rainer Schuler, Tomoyuki Yamakami*
Structural Average Case Complexity

95-05 *Klaus Achatz, Wolfram Schulte*
Architecture Indepentent Massive Parallelization of Divide-And-Conquer Algorithms

95-06 *Christoph Karg, Rainer Schuler*
Structure in Average Case Complexity

95-07 *P. Dadam, K. Kuhn, M. Reichert, T. Beuter, M. Nathe*
ADEPT: Ein integrierender Ansatz zur Entwicklung flexibler, zuverlässiger kooperierender Assistenzsysteme in klinischen Anwendungsumgebungen

95-08 *Jürgen Kehrer, Peter Schulthess*
Aufbereitung von gescannten Röntgenbildern zur filmlosen Diagnostik

95-09 *Hans-Jörg Burtschick, Wolfgang Lindner*
On Sets Turing Reducible to P-Selective Sets

95-10 *Boris Hartmann*
Berücksichtigung lokaler Randbedingung bei globaler Zieloptimierung mit neuronalen Netzen am Beispiel Truck Backer-Upper

95-12 *Klaus Achatz, Wolfram Schulte*
Massive Parallelization of Divide-and-Conquer Algorithms over Powerlists

95-13 *Andrea Mößle, Heiko Vogler*
Efficient Call-by-value Evaluation Strategy of Primitive Recursive Program Schemes

95-14 *Axel Dold, Friedrich W. von Henke, Holger Pfeifer, Harald Rueß*
A Generic Specification for Verifying Peephole Optimizations

96-01 *Ercüment Canver, Jan-Tecker Gayen, Adam Moik*
Formale Entwicklung der Steuerungssoftware für eine elektrisch ortsbediente Weiche mit VSE

96-02 *Bernhard Nebel*
Solving Hard Qualitative Temporal Reasoning Problems: Evaluating the Efficiency of Using the ORD-Horn Class

96-03 *Ton Vullinghs, Wolfram Schulte, Thilo Schwinn*
An Introduction to TkGofer

96-04 *Thomas Beuter, Peter Dadam*
Anwendungsspezifische Anforderungen an Workflow-Mangement-Systeme am Beispiel der Domäne Concurrent-Engineering

96-05 *Gerhard Schellhorn, Wolfgang Ahrendt*
Verification of a Prolog Compiler - First Steps with KIV

96-06 *Manindra Agrawal, Thomas Thierauf*
Satisfiability Problems

96-07 *Vikraman Arvind, Jacobo Torán*
A nonadaptive NC Checker for Permutation Group Intersection

96-08 *David Cyrluk, Oliver Möller, Harald Rueß*
An Efficient Decision Procedure for a Theory of Fix-Sized Bitvectors with Composition and Extraction

96-09 *Bernd Biechele, Dietmar Ernst, Frank Houdek, Joachim Schmid, Wolfram Schulte*
Erfahrungen bei der Modellierung eingebetteter Systeme mit verschiedenen SA/RT–Ansätzen

96-10 *Falk Bartels, Axel Dold, Friedrich W. von Henke, Holger Pfeifer, Harald Rueß*
Formalizing Fixed-Point Theory in PVS

96-11 *Axel Dold, Friedrich W. von Henke, Holger Pfeifer, Harald Rueß*
Mechanized Semantics of Simple Imperative Programming Constructs

96-12 *Axel Dold, Friedrich W. von Henke, Holger Pfeifer, Harald Rueß*
Generic Compilation Schemes for Simple Programming Constructs

96-13 *Klaus Achatz, Helmuth Partsch*
From Descriptive Specifications to Operational ones: A Powerful Transformation Rule, its Applications and Variants

97-01 *Jochen Messner*
Pattern Matching in Trace Monoids

97-02 *Wolfgang Lindner, Rainer Schuler*
A Small Span Theorem within P

97-03 *Thomas Bauer, Peter Dadam*
A Distributed Execution Environment for Large-Scale Workflow Management Systems with Subnets and Server Migration

97-04 *Christian Heinlein, Peter Dadam*
Interaction Expressions - A Powerful Formalism for Describing Inter-Workflow Dependencies

97-05 *Vikraman Arvind, Johannes Köbler*
On Pseudorandomness and Resource-Bounded Measure

97-06 *Gerhard Partsch*
Punkt-zu-Punkt- und Mehrpunkt-basierende LAN-Integrationsstrategien
für den digitalen Mobilfunkstandard DECT

97-07 *Manfred Reichert, Peter Dadam*
$ADEPT_{flex}$ - Supporting Dynamic Changes of Workflows Without Loosing Control

97-08 *Hans Braxmeier, Dietmar Ernst, Andrea Mößle, Heiko Vogler*
The Project NoName - A functional programming language with its development environment

97-09 *Christian Heinlein*
Grundlagen von Interaktionsausdrücken

97-10 *Christian Heinlein*
Graphische Repräsentation von Interaktionsausdrücken

97-11 *Christian Heinlein*
Sprachtheoretische Semantik von Interaktionsausdrücken

97-12 *Gerhard Schellhorn, Wolfgang Reif*
Proving Properties of Finite Enumerations: A Problem Set for Automated Theorem Provers

97-13 *Dietmar Ernst, Frank Houdek, Wolfram Schulte, Thilo Schwinn*
Experimenteller Vergleich statischer und dynamischer Softwareprüfung für eingebettete Systeme

97-14 *Wolfgang Reif, Gerhard Schellhorn*
Theorem Proving in Large Theories

97-15 *Thomas Wennekers*
Asymptotik rekurrenter neuronaler Netze mit zufälligen Kopplungen

97-16 *Peter Dadam, Klaus Kuhn, Manfred Reichert*
Clinical Workflows - The Killer Application for Process-oriented Information Systems?

97-17 *Mohammad Ali Livani, Jörg Kaiser*
EDF Consensus on CAN Bus Access in Dynamic Real-Time Applications

97-18 *Johannes Köbler,Rainer Schuler*
Using Efficient Average-Case Algorithms to Collapse Worst-Case Complexity Classes

98-01 *Daniela Damm, Lutz Claes, Friedrich W. von Henke, Alexander Seitz, Adelinde Uhrmacher, Steffen Wolf*
Ein fallbasiertes System für die Interpretation von Literatur zur Knochenheilung

98-02 *Thomas Bauer, Peter Dadam*
Architekturen für skalierbare Workflow-Management-Systeme - Klassifikation und Analyse

98-03 *Marko Luther, Martin Strecker*
A guided tour through *Typelab*

98-04 *Heiko Neumann, Luiz Pessoa*
Visual Filling-in and Surface Property Reconstruction

98-05 *Ercüment Canver*
Formal Verification of a Coordinated Atomic Action Based Design

98-06 *Andreas Küchler*
On the Correspondence between Neural Folding Architectures and Tree Automata

98-07 *Heiko Neumann, Thorsten Hansen, Luiz Pessoa*
Interaction of ON and OFF Pathways for Visual Contrast Measurement

98-08 *Thomas Wennekers*
Synfire Graphs: From Spike Patterns to Automata of Spiking Neurons

98-09 *Thomas Bauer, Peter Dadam*
Variable Migration von Workflows in *ADEPT*

98-10 *Heiko Neumann, Wolfgang Sepp*
Recurrent V1 – V2 Interaction in Early Visual Boundary Processing

98-11 *Frank Houdek, Dietmar Ernst, Thilo Schwinn*
Prüfen von C–Code und Statmate/Matlab–Spezifikationen: Ein Experiment

98-12 *Gerhard Schellhorn*
Proving Properties of Directed Graphs: A Problem Set for Automated Theorem Provers

98-13 *Gerhard Schellhorn, Wolfgang Reif*
Theorems from Compiler Verification: A Problem Set for Automated Theorem Provers

98-14 *Mohammad Ali Livani*
SHARE: A Transparent Mechanism for Reliable Broadcast Delivery in CAN

98-15 *Mohammad Ali Livani, Jörg Kaiser*
Predictable Atomic Multicast in the Controller Area Network (CAN)

99-01 *Susanne Boll, Wolfgang Klas, Utz Westermann*
A Comparison of Multimedia Document Models Concerning Advanced Requirements

99-02 *Thomas Bauer, Peter Dadam*
Verteilungsmodelle für Workflow-Management-Systeme - Klassifikation und Simulation

99-03 *Uwe Schöning*
On the Complexity of Constraint Satisfaction

99-04 *Ercument Canver*
Model-Checking zur Analyse von Message Sequence Charts über Statecharts

99-05 *Johannes Köbler, Wolfgang Lindner, Rainer Schuler*
Derandomizing RP if Boolean Circuits are not Learnable

99-06 *Utz Westermann, Wolfgang Klas*
Architecture of a DataBlade Module for the Integrated Management of Multimedia Assets

99-07 *Peter Dadam, Manfred Reichert*
Enterprise-wide and Cross-enterprise Workflow Management: Concepts, Systems, Applications. Paderborn, Germany, October 6, 1999, GI–Workshop Proceedings, Informatik '99

99-08 *Vikraman Arvind, Johannes Köbler*
Graph Isomorphism is Low for $\mathrm{ZPP}^{\mathrm{NP}}$ and other Lowness results

99-09 *Thomas Bauer, Peter Dadam*
Efficient Distributed Workflow Management Based on Variable Server Assignments

2000-02 *Thomas Bauer, Peter Dadam*
Variable Serverzuordnungen und komplexe Bearbeiterzuordnungen im Workflow-Management-System ADEPT

2000-03 *Gregory Baratoff, Christian Toepfer, Heiko Neumann*
Combined space-variant maps for optical flow based navigation

2000-04 *Wolfgang Gehring*
Ein Rahmenwerk zur Einführung von Leistungspunktsystemen

2000-05 *Susanne Boll, Christian Heinlein, Wolfgang Klas, Jochen Wandel*
Intelligent Prefetching and Buffering for Interactive Streaming of MPEG Videos

2000-06 *Wolfgang Reif, Gerhard Schellhorn, Andreas Thums*
Fehlersuche in Formalen Spezifikationen

2000-07 *Gerhard Schellhorn, Wolfgang Reif (eds.)*
FM-Tools 2000: The 4^{th} Workshop on Tools for System Design and Verification

2000-08 *Thomas Bauer, Manfred Reichert, Peter Dadam*
Effiziente Durchführung von Prozessmigrationen in verteilten Workflow-Management-Systemen

2000-09 *Thomas Bauer, Peter Dadam*
Vermeidung von Überlastsituationen durch Replikation von Workflow-Servern in ADEPT

2000-10 *Thomas Bauer, Manfred Reichert, Peter Dadam*
Adaptives und verteiltes Workflow-Management

2000-11 *Christian Heinlein*
Workflow and Process Synchronization with Interaction Expressions and Graphs

2001-01 *Hubert Hug, Rainer Schuler*
DNA-based parallel computation of simple arithmetic

2001-02 *Friedhelm Schwenker, Hans A. Kestler, Günther Palm*
3-D Visual Object Classification with Hierarchical Radial Basis Function Networks

2001-03 *Hans A. Kestler, Friedhelm Schwenker, Günther Palm*
RBF network classification of ECGs as a potential marker for sudden cardiac death

2001-04 *Christian Dietrich, Friedhelm Schwenker, Klaus Riede, Günther Palm*
Classification of Bioacoustic Time Series Utilizing Pulse Detection, Time and Frequency Features and Data Fusion

2002-01 *Stefanie Rinderle, Manfred Reichert, Peter Dadam*
Effiziente Verträglichkeitsprüfung und
automatische Migration von Workflow-Instanzen
bei der Evolution von Workflow-Schemata

2002-02 *Walter Guttmann*
Deriving an Applicative Heapsort Algorithm

2002-03 *Axel Dold, Friedrich W. von Henke, Vincent Vialard, Wolfgang Goerigk*
A Mechanically Verified Compiling Specification for a Realistic Compiler

2003-01 *Manfred Reichert, Stefanie Rinderle, Peter Dadam*
A Formal Framework for Workflow Type and Instance Changes Under Correctness Checks

2003-02 *Stefanie Rinderle, Manfred Reichert, Peter Dadam*
Supporting Workflow Schema Evolution By Efficient Compliance Checks

2003-03 *Christian Heinlein*
Safely Extending Procedure Types to Allow Nested Procedures as Values

2003-04 *Stefanie Rinderle, Manfred Reichert, Peter Dadam*
On Dealing With Semantically Conflicting Business Process Changes.

2003-05 *Christian Heinlein*
Dynamic Class Methods in Java

2003-06 *Christian Heinlein*
Vertical, Horizontal, and Behavioural Extensibility of Software Systems

2003-07 *Christian Heinlein*
Safely Extending Procedure Types to Allow Nested Procedures as Values
(Corrected Version)

2003-08 *Changling Liu, Jörg Kaiser*
Survey of Mobile Ad Hoc Network Routing Protocols)

2004-01 *Thom Frühwirth, Marc Meister (eds.)*
First Workshop on Constraint Handling Rules

2004-02 *Christian Heinlein*
Concept and Implementation of C+++, an Extension of C++ to Support User-Defined Operator Symbols and Control Structures

2004-03 *Susanne Biundo, Thom Frühwirth, Günther Palm(eds.)*
Poster Proceedings of the 27th Annual German Conference on Artificial Intelligence

2005-01 *Armin Wolf, Thom Frühwirth, Marc Meister (eds.)*
19th Workshop on (Constraint) Logic Programming

2005-02 *Wolfgang Lindner (Hg.), Universität Ulm , Christopher Wolf (Hg.) KU Leuven*
2. Krypto-Tag – Workshop über Kryptographie, Universität Ulm

2005-03 *Walter Guttmann, Markus Maucher*
Constrained Ordering

2006-01 *Stefan Sarstedt*
Model-Driven Development with ACTIVECHARTS, Tutorial

2006-02 *Alexander Raschke, Ramin Tavakoli Kolagari*
Ein experimenteller Vergleich zwischen einer plan-getriebenen und einer leichtgewichtigen Entwicklungsmethode zur Spezifikation von eingebetteten Systemen

2006-03 *Jens Kohlmeyer, Alexander Raschke, Ramin Tavakoli Kolagari*
Eine qualitative Untersuchung zur Produktlinien-Integration über Organisationsgrenzen hinweg

2006-04 *Thorsten Liebig*
Reasoning with OWL
- System Support and Insights -

Ulmer Informatik-Berichte

ISSN 0939-5091

Herausgeber: Fakultät für Informatik

Universität Ulm, Oberer Eselsberg, D-89069 Ulm