Deutsches
Forschungszentrum
für Künstliche
Intelligenz GmbH

Research Report
RR-92-16

An Empirical Analysis of Terminological Representation Systems

**Jochen Heinsohn, Daniel Kudenko,
Bernhard Nebel, Hans-Jürgen Profitlich**

May 1992

Deutsches Forschungszentrum für Künstliche Intelligenz GmbH

Postfach 20 80
D-6750 Kaiserslautern, FRG
Tel.: (+49 631) 205-3211/13
Fax: (+49 631) 205-3210

Stuhlsatzenhausweg 3
D-6600 Saarbrücken 11, FRG
Tel.: (+49 681) 302-5252
Fax: (+49 681) 302-5341

Deutsches Forschungszentrum für Künstliche Intelligenz

The German Research Center for Artificial Intelligence (Deutsches Forschungszentrum für Künstliche Intelligenz, DFKI) with sites in Kaiserslautern and Saarbrücken is a non-profit organization which was founded in 1988. The shareholder companies are Atlas Elektronik, Daimler Benz, Fraunhofer Gesellschaft, GMD, IBM, Insiders, Mannesmann-Kienzle, Philips, SEMA Group Systems, Siemens and Siemens-Nixdorf. Research projects conducted at the DFKI are funded by the German Ministry for Research and Technology, by the shareholder companies, or by other industrial contracts.

The DFKI conducts application-oriented basic research in the field of artificial intelligence and other related subfields of computer science. The overall goal is to construct *systems with technical knowledge and common sense* which - by using AI methods - implement a problem solution for a selected application area. Currently, there are the following research areas at the DFKI:

- Intelligent Engineering Systems
- Intelligent User Interfaces
- Intelligent Communication Networks
- Intelligent Cooperative Systems.

The DFKI strives at making its research results available to the scientific community. There exist many contacts to domestic and foreign research institutions, both in academy and industry. The DFKI hosts technology transfer workshops for shareholders and other interested groups in order to inform about the current state of research.

From its beginning, the DFKI has provided an attractive working environment for AI researchers from Germany and from all over the world. The goal is to have a staff of about 100 researchers at the end of the building-up phase.

Prof. Dr. Gerhard Barth
Director

An Empirical Analysis of Terminological Representation Systems

**Jochen Heinsohn, Daniel Kudenko,
Bernhard Nebel, Hans-Jürgen Profitlich**

DFKI-RR-92-16

A short version of this report will be published in the Proceedings of the Tenth National Conference on Artificial Intelligence (AAAI-92), July 12 - 17, 1992, San Jose, California

This work has been supported by a grant from The Federal Ministry for Research and Technology (FKZ ITW 8901 8).

An Empirical Analysis of Terminological Representation Systems*

Jochen Heinsohn, Daniel Kudenko,
Bernhard Nebel, and Hans-Jürgen Profitlich

German Research Center for Artificial Intelligence (DFKI)
Stuhlsatzenhausweg 3
W-6600 Saarbrücken, Germany
e-mail: ⟨*last name*⟩@dfki.uni-sb.de

Abstract

The family of terminological representation systems has its roots in the representation system KL-ONE. Since the development of this system more than a dozen similar representation systems have been developed by various research groups. These systems vary along a number of dimensions. In this paper, we present the results of an empirical analysis of six such systems. Surprisingly, the systems turned out to be quite diverse leading to problems when transporting knowledge bases from one system to another. Additionally, the runtime performance between different systems and knowledge bases varied more than we expected. Finally, our empirical runtime performance results give an idea of what runtime performance to expect from such representation systems. These findings complement previously reported analytical results about the computational complexity of reasoning in such systems.

*This work has been carried out in the WIP project which is supported by the German Ministry for Research and Technology BMFT under contract ITW 8901 8.

Contents

1 Introduction **1**

2 The Experiment **2**

3 Systems **3**

4 Qualitative Results **4**

5 Quantitative Results **6**

6 Conclusions **11**

A The Common Terminological Language CTL **16**

B Expressiveness of the Systems **20**

C Problematic Cases **23**

D TBox Inferences **27**

E Hard Cases **31**

F Real Knowledge Bases **34**

G Random Knowledge Bases **37**

List of Tables

1 CTL: Concept forming operators 17
2 CTL: Role and attribute forming operators 18
3 CTL: Additional restrictive operators 18
4 CTL: Terminological axioms 19
5 Expressiveness: Concept forming operators 20
6 Expressiveness: Role and attribute forming operators 21
7 Expressiveness: Additional restrictive operators 21
8 Expressiveness: Terminological axioms 22
9 Problematic Cases: Alternative constructs 23
10 Problematic Cases: Test results 26
11 TBox Inferences: Test results 30
12 Hard Cases: Test results 33
13 Real Knowledge Bases: Structural description 35
14 Real Knowledge Bases: Test results 36
15 Random knowledge bases: Test results 38

List of Figures

1 Experiment design 2
2 Runtime performance for hard cases 7
3 Runtime performance for realistic cases 8
4 Runtime performance for small random KBs 9
5 Runtime performance for large random KBs 10
6 Runtime performance for very large random KBs 11

1 Introduction

Terminological representation systems support the taxonomic representation of terminology for AI applications and provide reasoning services over the terminology. Such systems may be used as stand-alone information retrieval systems [11] or as components of larger AI systems, such as natural language systems [30] or design systems [32]. Assuming that the application task is the configuration of computer systems [23], the terminology may contain *concepts* such as *local area network, workstation, disk-less workstation, file server*, etc. Further, these concepts are interrelated by specialization relationships and the specification of necessary and sufficient conditions. A *disk-less workstation* may be *defined* as a *workstation* that has no *disk* attached to it, for example. The main reasoning service provided by terminological representation systems is checking for inconsistencies in concept specifications and determining the specialization relation between concepts—the so-called *subsumption relation.*

The first knowledge representation system supporting this kind of representation and reasoning was KL-ONE [8]. Meanwhile, the underlying framework has been adopted by various research groups, and more than a dozen terminological representation systems have been implemented [25]. These systems vary along a number of important dimensions, such as implementation status, expressiveness of the underlying representation language, completeness of the reasoning services, efficiency, user interface, interface functionality, and integration with other modes of reasoning.

Nowadays, it seems reasonable to build upon an existing terminological representation system instead of building one from scratch. Indeed, this was the idea in our project WIP, which is aimed at knowledge-based, multi-modal presentation of information such as operating instructions [31]. However, it was by no means clear which system to choose. For this reason, we analyzed a subset of the available systems empirically. It turned out that the effort we had to invest could have well been used to implement an additional prototypical terminological representation system. However, we believe that the experience gained is worthwhile, in particular concerning the implementation of future terminological representation systems and standard efforts in the area of terminological representation systems.

One of the main results of our study is that the differences in expressiveness between the existing systems are larger than one would expect considering the fact that all of them are designed using a common semantic framework. These differences led to severe problems when we transported knowledge bases between the systems. Another interesting result is the runtime performance data we obtained. These findings indicate (1) that the structure of the knowledge base can have a significant impact on the performance, (2) that the runtime grows faster than linearly in all systems, and (3) that implementations ignoring efficiency issues can be quite slow. Additionally, the performance data gives an idea of what performance to expect from existing terminological representation systems. These results complement the various analytical results on the computational complexity of terminological reasoning.

The rest of the paper is structured as follows. In the next section we give a sketch of the experiment design, and in Section 3 we briefly describe the systems we analyzed. The qualitative results are given in Section 4, and the quantitative results are described in Section 5. The details of the experiments and their results are described in the appendices.

2 The Experiment

The empirical analysis can be roughly divided into two parts (see Figure 1). The first part covers qualitative facts concerning system features and expressiveness. In order to describe the latter aspect, we first developed a "common terminological language" that covers a superset of all terminological languages employed in the systems we considered (see also Appendix A and [3]). The analysis of the expressiveness shows that the intersection over all representation languages used in the systems is quite small.

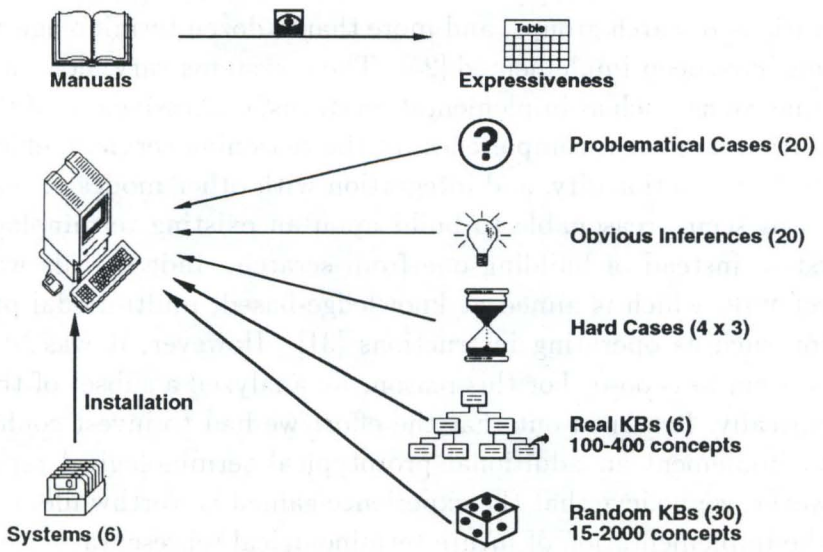

Figure 1: Experiment design

In the second part we ran different test cases on the systems in order to check out the performance, completeness and the handling of problematical cases. We designed five different groups of experiments. The first group consists of tests dealing with cases that are not covered by the common semantic framework of terminological representation systems. The second group explores the degree of the inferential completeness of the systems for "easy" (i.e., polynomial) inferences. It should be noted that we did not try to design these tests in a systematic fashion by trying out all possible combinations of language constructs, though. The third group consists

of problems which are known to be "hard" for existing systems. They give an impression of the runtime performance under worst-case conditions.

For the fourth group of experiments we used existing knowledge bases to get an idea of the runtime performance under "realistic" conditions. First, we manually converted the knowledge bases into the "common terminological language" mentioned above. Then, we implemented a number of translators that map the knowledge bases formulated using the "common terminological language" into system specific knowledge bases.

Although the results of the fourth group of experiments give some clues of what the behavior of the systems may be in applications, we had not enough data points to confirm some of the conjectures that resulted from this initial test under realistic conditions. Additionally, it was not evident in how far the translation, which is only approximate, influenced the performance. For this reason, a fifth group of experiments was designed. A number of knowledge bases were generated randomly with a structure similar to the structure of the realistic knowledge bases.

In general, we concentrated on the *terminological representation* part (also called *TBox*) of the systems. This means that we ignored other representation and reasoning facilities, such as facilities for maintaining and manipulating databases of objects (also called *ABox*) that are described by using the concepts represented in the terminological knowledge base. This concentration on the terminological component is partly justified by the fact that the terminological part is the one which participates in most reasoning activities of the entire system. Thus, runtime performance and completeness of the terminological part can be generalized to the entire system—to a certain degree. However, the systems may (for efficiency reasons) use different algorithms for maintaining a database of objects, which may lead to a different behavior in this case. Nevertheless, even if the generalization is not valid in general, we get at least a feeling how the terminological parts perform.

As a final note, we want to emphasize that our empirical analysis was not intended to establish a ranking between the systems. For this purpose, it would be necessary to assign weights to the dimensions we compared, and this can only be done if the intended application has been fixed. Despite the fact that we analyzed only the terminological subsystems, the tests are not intended to be complete in any sense and there may be more dimensions that could be used to analyze the systems. Further, the results apply, of course, only to the system versions explicitly mentioned in the following section. The system developers of a number of systems have improved their systems since we made our experiment. So, the runtime performance may have changed.

3 Systems

There are a large number of systems which could have been included in an empirical analysis, e.g., KL-ONE [8], LILOG [5], NIKL [29], K-REP [19], KRS [14], KRYPTON [7], YAK [9]. However, we concentrated on a relatively small number of systems. This

does not mean that we feel that the systems we did not include (or mention) are not worthwhile to be analyzed. The only reason not to include all the systems was the limited amount of time available. We hope, however, that our investigation can serve as a starting point for future empirical analyses. The systems we picked for the experiment are: BACK [26] (Version 4.2, pre-released), CLASSIC [24] (Version 1.02, released), KRIS [4] (Version 1.0, experimental), LOOM [18] (Version of May 1990, pre-released), MESON [23] (Version 2.0, released), and SB-ONE [16] (Version of January 1990, released).

The BACK system has been developed at the Technical University of Berlin by the KIT-BACK group as part of the Esprit project ADKMS. The main application is an information system about the financial and organizational structure of a company [10]. It is the only system among the ones we tested that is written in PROLOG. We tested the system on a Solbourne 601/32 using SICSTUS-PROLOG 2.1.

CLASSIC has been developed in the AI Principles Research Department at AT&T Bell Laboratories. It supports only a very limited terminological language, but turned out to be very useful for a number of applications [11]. As all other systems except for BACK, it is written in COMMONLISP and we tested it on a MacIvory.

KRIS has been developed by the WINO project at DFKI. In contrast to other systems, it provides complete inference algorithms for very expressive languages. Efficiency considerations have played no role in the development of the system.

LOOM has been developed at USC/ISI and supports a very powerful terminological logic—in an incomplete manner, though—and offers the user a very large number of features. In fact, LOOM can be considered as a programming environment.

MESON has been developed at the Philips Research Laboratories, Hamburg, as a KR tool for different applications, e.g., computer configuration [23]. Although it is also written in COMMONLISP, we tested it not on a MacIvory but on a Solbourne 601/32 in order to take advantage of its nice X-Window interface.

SB-ONE has been developed in the XTRA project at the University of Saarland as the knowledge representation tool for a natural language project. One of the main ideas behind the design of the system was the possibility of direct graphical manipulations of the represented knowledge.

4 Qualitative Results

The main qualitative result of our experiment is that although the systems were developed with a common framework in mind, they are much more diverse than one would expect. First of all, the terminological languages that are supported by the various systems are quite different. While three of the six systems use a similar syntactic scheme (similar to the one first used by Brachman and Levesque [6]), and one system adapted this syntactic scheme for PROLOG, i.e., infix instead of prefix notation, the remaining two systems use quite different syntactic schemes. Furthermore, there are not only superficial differences in the syntax, but the set of (underlying) *term-forming* operators varies, as well. In fact, the common intersection of all

languages we considered is quite small. It contains only the concept-forming operators *concept conjunction*, *value restriction*, and *number restriction*. The detailed description of the expressiveness of the different systems is given in Appendix B.

These differences led to severe problems when we designed automatic translators from the "common terminological language" to the languages supported by the different systems. Because of the differences in expressiveness, the translations could only be approximate, and because of the differences in the syntax we used a translation schema that preserved the meaning (as far as possible) but introduced a number of auxiliary concepts. Using the translated knowledge bases, we noticed that the introduction of auxiliary concepts (see Appendix F) influences the runtime performance significantly—a point we will return to in Section 5.

Discounting the differences in syntax and expressiveness, one might expect that the common semantic framework (as spelled out by Brachman and Levesque [6]) leads to identical behavior on inputs that have identical meaning and match the expressiveness of the systems. However, this is unfortunately wrong. When a formal specification is turned into an implemented system, there are a number of areas that are not completely covered by the specification. One example is the order of the input. So, some systems allow for *forward references* in term definitions and some do not. Furthermore, some systems support *cyclic definitions* (without handling them correctly according to one of the possible semantics [21], however, or permitting cyclic definitions only in some contexts), and some give an error message. Also *redefinitions* of terms are either marked as errors, processed as revisions of the terminology, or treated as incremental additions to the definition. Finally, there are different rules for *determining the syntactic category* of an input symbol.

Another area where designers of terminological systems seem to disagree is what should be considered as an error by the user. So, some systems mark the definitions of *semantically equivalent* concepts as an error or refuse to accept *semantically empty* (inconsistent) concepts, for instance (see Appendix C).

These differences between the systems made the translation from the "common terminological language" to system-specific languages even more complicated. In fact, some of the problems mentioned above were only discovered when we ran the systems on the translated knowledge bases. We solved that problem by putting the source form of the knowledge base into the most unproblematical form, if possible, or ignored problematical constructions (such as cyclic definitions) in the translation process.

Summarizing, these results show that the ongoing process of specifying a common language for terminological representation and reasoning systems [22, p. 50–51] will probably improve the situation in so far as the translation of knowledge bases between different systems will become significantly easier. One main point to observe, however, is the area of pragmatics we touched above, such as permitting forward references.

Finally, we should mention a point which all systems had in common. In each system we discovered at least one deviation from the documentation, such as missing

an obvious inference or giving a wrong error message. This is, of course, not surprising, but shows that standard test suites should be developed for these systems.

There are a number of other dimensions where the systems differ, such as the integration with other reasoning services, the functionality of graphical user interfaces, ease of installation, and user friendliness, but these are issues which are very difficult to evaluate.

5 Quantitative Results

One important feature of a representation and reasoning system is, of course, its runtime performance. In the case of terminological representation systems, the time to compute the *subsumption hierarchy* of concepts—a process that is often called *classification*—is an interesting parameter. In order to get a feeling for the runtime behavior of the systems we designed several tests to explore how the systems behave under different conditions. Since most of the systems are still under development, the runtime data we gathered is most probably not an accurate picture of the performance of the most recent versions of the systems. In particular, new versions of BACK, CLASSIC, KRIS, and LOOM are available that are faster and/or support more expressive languages.

Computational complexity results show that *subsumption determination* between *terms* is NP-hard [13] or even undecidable [28] for reasonably expressive languages. Even assuming that *term-subsumption* can be computed in polynomial time (e.g., for restricted languages), subsumption determination in a *terminology* is still NP-hard [20]. In order to explore this issue, we designed some tests to determine the behavior of the systems under conditions that are known to be hard (see Appendix E).

One test exploits the NP-hardness result for term-subsumption for languages that contain concept-conjunction, value restrictions, and qualified existential restrictions [12]. It turned out that three systems could not express this case, one system reported an internal error, one system missed the inference (but exhibited a polynomial runtime behavior), and only one system handled the case, but with a very rapid growth in runtime.

Three other tests exploit the NP-hardness result for subsumption in terminologies [20]. The first two tests show that only one of the six systems uses a naive way of performing subsumption in a terminology by expanding all concept definitions before checking subsumption [20, p. 239]. The third test was designed in a way such that also clever subsumption algorithms are bound to use exponential time [20, p. 245].[1] The results of the latter test are given in Figure 2.[2] They clearly indicate that the

[1] Note that the example [20, p. 245] is not correct. The second appearance of each role R in all definitions should be R'. Further note that the number of concepts specified in Figure 2 does not coincide with the parameter n in the worst-case example [20, p. 245].

[2] The runtimes of BACK and MESON are not directly comparable with the other systems because BACK and MESON were tested on a Solbourne 601/32, which is two to three times faster than a MacIvory with respect to the execution of COMMONLISP programs, a remark that applies also to the other runtime performance tests. Additionally, it is not clear to us in how far the performance

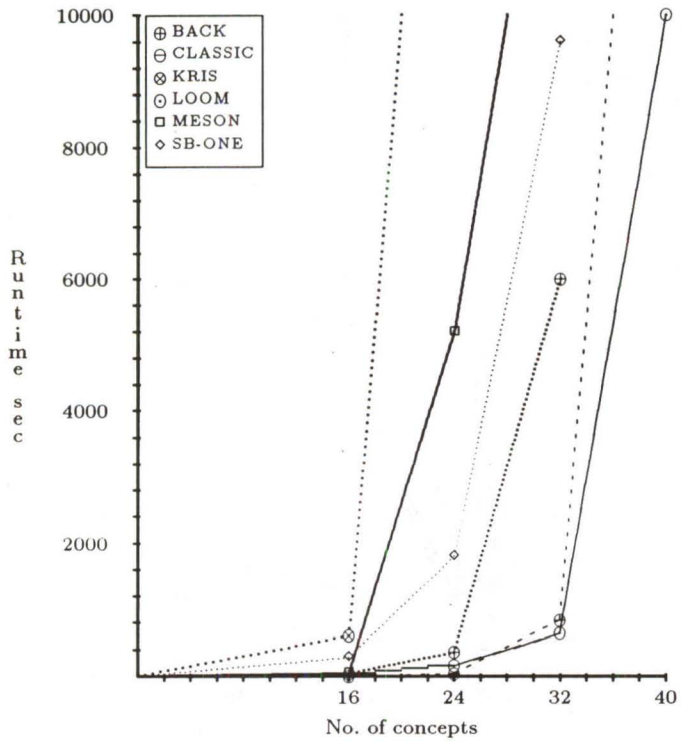

Figure 2: Runtime performance for hard cases

systems indeed exhibit a very rapid growth in the runtime.

Despite their theoretical intractability, terminological reasoning systems have been used for quite a while and the literature suggests that the knowledge bases involved were larger than just toy examples (i.e., more than 40 concepts). Hence, one would assume that the knowledge bases that have been used in applications are of a form that permits easy inferences, or the systems are incomplete and ignore costly inferences. In any case, it is questionable of whether the runtime performance for worst-case examples give us the right idea of how systems will behave in applications.

In order to get a feeling of the runtime performance under realistic conditions, we asked other research groups for terminological knowledge bases they use in their projects. Doing so, we obtained six different knowledge bases. As mentioned above, these were first manually translated into the "common terminological language" and then translated to each target language using our (semi-) automatic translators. In Figure 3, the runtime for the systems is plotted against the number of concepts defined in the different knowledge bases (see Appendix F).[3]

There are a number of interesting points to note here. First of all, two systems, namely, KRIS and SB-ONE, were too slow to be plotted together with the other systems using the same scale. For this reason, we divided the runtimes by the factor

of BACK is influenced by the fact that it is implemented in PROLOG.

[3]The number of concepts were counted *after* the translation, i.e., this number includes the auxiliary concepts introduced in the translation process.

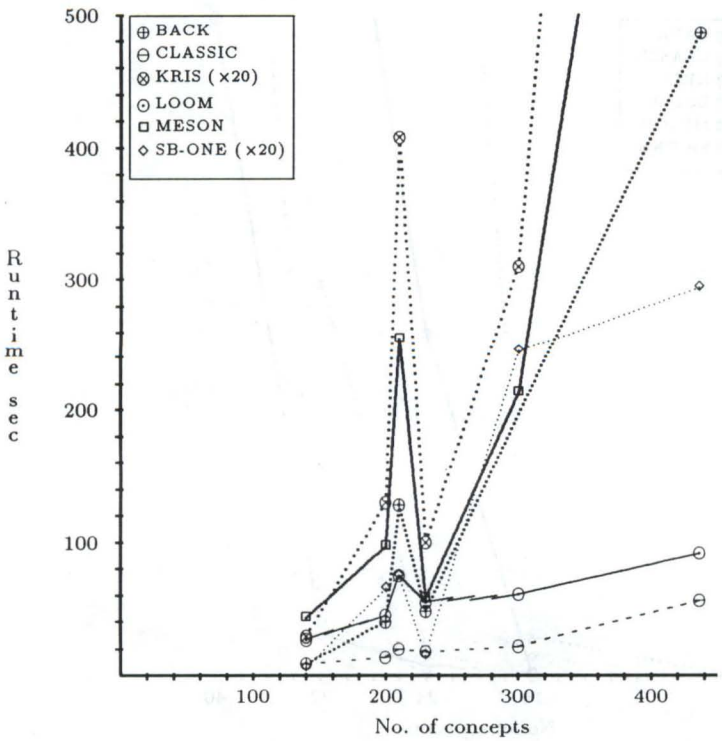

Figure 3: Runtime performance for realistic cases

of 20 before plotting it.

Second, the diagram indicates that the runtime ratio between the slowest system (KRIS) and the fastest system (CLASSIC) in case of the largest knowledge base is extreme, namely, $45,000/56 \approx 800$.[4] Considering that KRIS was developed as an experimental testbed for different complete subsumption algorithms and CLASSIC was designed as an efficient system for an expressively limited language to be used in different applications, this result is actually not completely surprising. It would be of course desirable to explain this and other differences in performance on the level of algorithms and implementation techniques. However, these issues are not described in the literature and a source code analysis was beyond the scope of our analysis.

Third, the knowledge base with 210 concepts seems to be somehow special because the runtime curve shows a peak at this point. Inspecting this knowledge base, we discovered that one concept is declared to be super-concept (i.e, mentioned literally in the definition) of 50% of all other concepts. Removing this concept led to a smoother curve. Hence, the structure of a knowledge base can severely influence

[4]This result cannot be attributed to the incompleteness of CLASSIC, the approximate character of the translation, or the differing number of auxiliary concepts introduced in the translation process. The knowledge base was formulated in a terminological language such that both CLASSIC and KRIS are complete, which implies that in this case the translation was also meaning preserving. Further, the knowledge base for KRIS contains fewer auxiliary concepts than the knowledge base for CLASSIC.

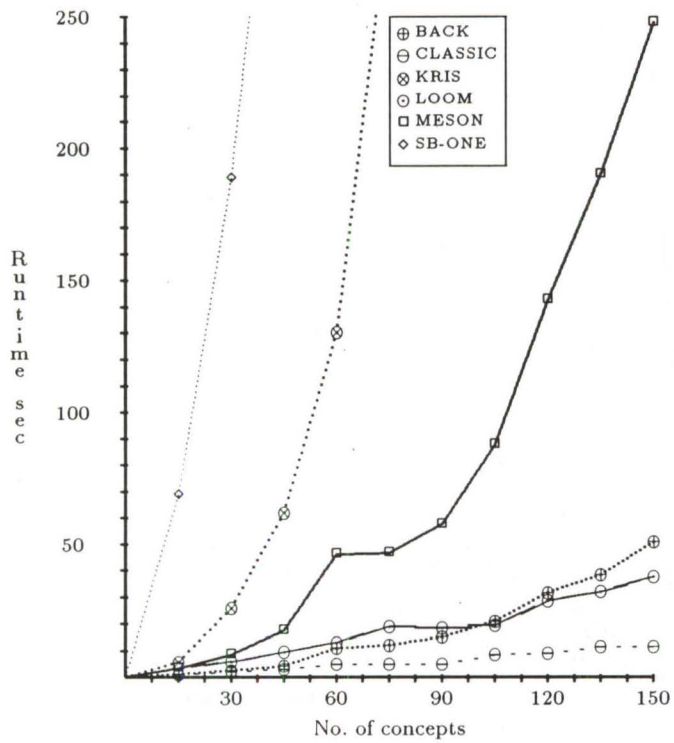

Figure 4: Runtime performance for small random KBs

the runtime. Although this should have been obvious already from the first diagram showing the runtime behavior under worst-case conditions, it is an indication that under realistic conditions the runtime behavior can be unexpectedly influenced by the structure of the knowledge base.

Summarizing the curves in Figure 3, it seems to be the case that most of the systems, except for SB-ONE, are similar in their runtime behavior in that the same knowledge bases are considered as "difficult" or "easy" to a similar degree. However, it is not clear whether the system runtimes differ only by a constant factor or not. Further, because of the approximative nature of the translations and the introduction of auxiliary concepts, it is not clear to us how reliable the data is. For these reasons, we generated knowledge bases randomly in the intersection of all languages—avoiding the translation problem. The structure of these generated knowledge bases resembles the structure of the six real knowledge bases (percentage of defined concepts, average number of declared super-concepts, average number of role restrictions, etc.). The results of this test are given in Figures 4, 5, and 6 (see also Appendix G).

Comparing the curves in these three figures with the curves in Figure 3, it seems to be the case that the structure of the randomly generated knowledge bases is indeed similar to the structure of realistic knowledge bases in so far as they lead to a similar runtime performance. However, we do not claim that the knowledge bases are realistic with respect to all possible aspects. In fact, too few facts are known

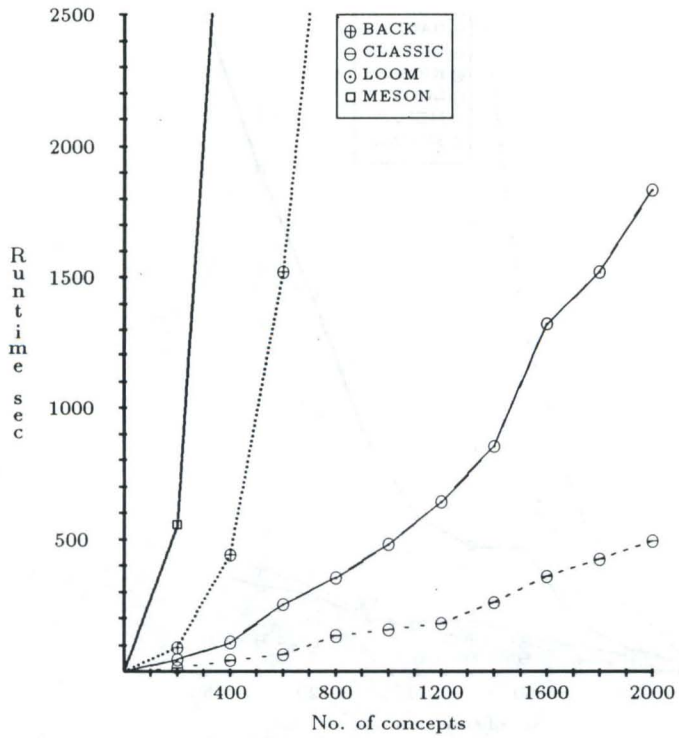

Figure 5: Runtime performance for large random KBs

about which structural properties can influence the performance of terminological representation systems. Bob MacGregor, for instance, reported that the number of distinct roles heavily influence the performance. He observed that the runtime *decreases* when the number of distinct roles is increased and all other parameters are hold constant (same number of concepts and role restrictions).

These curves indicate that the runtime grows faster than linearly with the number of concepts. We conjecture that in general the runtime of terminological representation systems is at least quadratic in the number of concepts. This conjecture is reasonable because identifying a partial order over a set of elements that are ordered by an underlying partial order is worst-case quadratic (if all elements are incomparable), and there is no algorithm known that is better for average cases. In fact, average case results are probably very hard to obtain because it is not known how many partial orders exist for a given number of elements [1, p. 271].

From this, we conclude that designing efficient terminological representation systems is not only a matter of designing efficient *subsumption algorithms*, but also a matter of designing efficient *classification algorithms*, i.e., fast algorithms that construct a partial order. The main point in this context is to minimize the number of subsumption tests.

Another conclusion of our runtime tests could be that the more expressive and complete a system is, the slower it is—with KRIS as a system supporting complete inferences for a very expressive language and CLASSIC with almost complete infer-

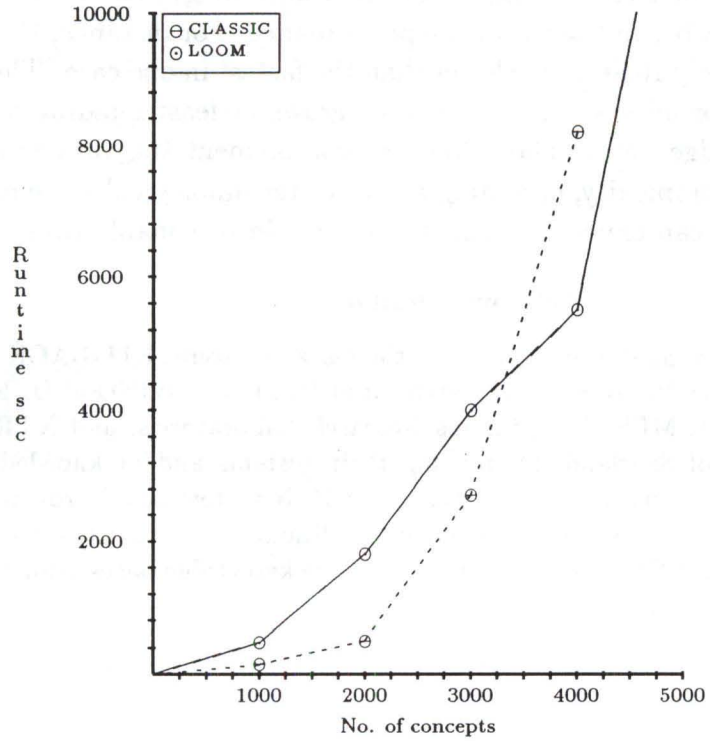

Figure 6: Runtime performance for very large random KBs

ences for a comparably simple language at the extreme points. However, we do not believe that this is a *necessary* phenomenon. A desirable behavior of such systems is that the user would have "to pay only as s/he goes," i.e., only if the full expressive power is used, the system is slow. In fact, together with the WINO group we are currently working on identifying the performance bottlenecks in the KRIS system. First experiences indicate that it is possible to come close to the performance of LOOM and CLASSIC for the knowledge bases used in our tests.

6 Conclusions

We have analyzed six different terminological representation and reasoning systems from a qualitative and quantitative point of view. The empirical analysis of the different terminological languages revealed that the common intersection of the languages supported by the systems is quite small. Together with the fact that the systems behave differently in areas that are not covered by the common semantic framework, sharing of knowledge bases between the systems does not seem to be easily achievable. In fact, when we tried to translate six different knowledge bases from a "common terminological language" into the system-specific languages we encountered a number of problems.

Testing the runtime performance of the systems, we noted that the structure of the knowledge base can have a significant impact on the performance, even if we do

not consider artificial worst-case examples but real knowledge bases. Further, the systems varied considerably in their runtime performance. For instance, the slowest system was approximately 1000 times slower than the fastest in one case. The overall picture suggests that for all systems the runtime grows at least quadratically with the size of the knowledge base. These findings complement the various analyses of the computational complexity, providing a user of terminological systems with a feeling of how much he can expect from such a system in reasonable time.

Acknowledgment

We would like to thank the members of the research groups KIT-BACK at TU Berlin, AI Principles Research Department at Bell Labs., WINO at DFKI, LOOM at USC/ISI, MESON at Philips Research Laboratories, and XTRA at the University of Saarland, for making their systems and/or knowledge bases available to us, answering questions about their systems, and providing comments on an earlier version of this paper. Additionally we want to express our thanks to Michael Gerlach, who made one of the knowledge bases available to us.

References

[1] M. Aigner. *Combinatorical Search.* Teubner, Stuttgart, Germany, 1988.

[2] H. Aït-Kaci. *A Lattice-Theoretic Approach to Computations Based on a Calculus of Partially Ordered Type Structures.* PhD thesis, University of Pennsylvania, Philadelphia, PA, 1984.

[3] F. Baader, H.-J. Bürckert, J. Heinsohn, B. Hollunder, J. Müller, B. Nebel, W. Nutt, and H.-J. Profitlich. Terminological Knowledge Representation: A proposal for a terminological logic. DFKI Report TM-90-04, Saarbrücken, 1990.

[4] F. Baader and B. Hollunder. KRIS: Knowledge representation and inference system. *SIGART Bulletin*, 2(3):8–14, June 1991.

[5] T. Bollinger and U. Pletat. The LILOG knowledge representation system. *SIGART Bulletin*, 2(3):22–27, June 1991.

[6] R. J. Brachman and H. J. Levesque. The tractability of subsumption in frame-based description languages. In *Proceedings of the 4th National Conference of the American Association for Artificial Intelligence*, pages 34–37, Austin, TX, Aug. 1984.

[7] R. J. Brachman, V. Pigman Gilbert, and H. J. Levesque. An essential hybrid reasoning system: Knowledge and symbol level accounts in KRYPTON. In *Proceedings of the 9th International Joint Conference on Artificial Intelligence*, pages 532–539, Los Angeles, CA, Aug. 1985.

[8] R. J. Brachman and J. G. Schmolze. An overview of the KL-ONE knowledge representation system. *Cognitive Science*, 9(2):171–216, Apr. 1985.

[9] R. Cattoni and E. Franconi. Walking through the semantics of frame-based description languages: A case study. In *Proceedings of the Fifth International Symposium on Methodologies for Intelligent systems*, Knoxville, TN, Oct. 1990. North-Holland.

[10] M. Damiani, S. Bottarelli, M. Migliorati, and C. Peltason. Terminological Information Management in ADKMS. In *ESPRIT '90 Conference Proceedings*, Dordrecht, Holland, 1990. Kluwer.

[11] P. T. Devanbu, R. J. Brachman, P. G. Selfridge, and B. W. Ballard. LaSSIE: a knowledge-based software information system. *Communications of the ACM*, 34(5):35–49, May 1991.

[12] F. M. Donini, M. Lenzerini, D. Nardi, B. Hollunder, W. Nutt, and A. M. Spacamella. The complexity of existential quantification in concept languages. *Artificial Intelligence* 53:309–327, 1992.

[13] F. M. Donini, M. Lenzerini, D. Nardi, and W. Nutt. The complexity of concept languages. In J. A. Allen, R. Fikes, and E. Sandewall, editors, *Principles of Knowledge Representation and Reasoning: Proceedings of the 2nd International Conference*, pages 151–162, Cambridge, MA, Apr. 1991.

[14] B. R. Gaines. Empirical investigations of knowledge representation servers: Design issues and application experience with KRS. *SIGART Bulletin*, 2(3):45–56, June 1991.

[15] B. Hollunder and W. Nutt. Subsumption algorithms for concept languages. DFKI Research Report RR-90-04, German Research Center for Artificial Intelligence (DFKI), Kaiserslautern, 1990.

[16] A. Kobsa. First experiences with the SB-ONE knowledge representation workbench in natural-language applications. *SIGART Bulletin*, 2(3):70–76, June 1991.

[17] H. J. Levesque and R. J. Brachman. Expressiveness and tractability in knowledge representation and reasoning. *Computational Intelligence*, 3:78–93, 1987.

[18] R. MacGregor. Inside the LOOM description classifier. *SIGART Bulletin*, 2(3):88–92, June 1991.

[19] E. Mays, R. Dionne, and R. Weida. K-Rep system overview. *SIGART Bulletin*, 2(3):93–97, June 1991.

[20] B. Nebel. Terminological reasoning is inherently intractable. *Artificial Intelligence*, 43:235–249, 1990.

[21] B. Nebel. Terminological cycles: Semantics and computational properties. In J. Sowa, editor, *Principles of Semantic Networks*, pages 331–362. Morgan Kaufmann, San Mateo, CA, 1991.

[22] R. Neches, R. Fikes, T. Finin, T. Gruber, R. Patil, T. Senator, and W. R. Swartout. Enabling technology for knowledge sharing. *The AI Magazine*, 12(3):36–56, 1991.

[23] B. Owsnicki-Klewe. Configuration as a consistency maintenance task. In W. Hoeppner, editor, *Künstliche Intelligenz. GWAI-88, 12. Jahrestagung*, pages 77–87. Springer-Verlag, Berlin, Heidelberg, New York, 1988.

[24] P. F. Patel-Schneider, D. L. McGuinness, R. J. Brachman, L. Alperin Resnick, and A. Borgida. The CLASSIC knowledge representation system: Guiding principles and implementation rational. *SIGART Bulletin*, 2(3):108–113, June 1991.

[25] P. F. Patel-Schneider, B. Owsnicki-Klewe, A. Kobsa, N. Guarino, R. MacGregor, W. S. Mark, D. McGuinness, B. Nebel, A. Schmiedel, and J. Yen. Term subsumption languages in knowledge representation. *The AI Magazine*, 11(2):16–23, 1990.

[26] C. Peltason. The BACK system – an overview. *SIGART Bulletin*, 2(3):114–119, June 1991.

[27] C. Peltason, A. Schmiedel, C. Kindermann, and J. Quantz. The BACK system revisited. KIT Report 75, Department of Computer Science, Technische Universität Berlin, Berlin, Germany, Sept. 1989.

[28] M. Schmidt-Schauß. Subsumption in KL-ONE is undecidable. In R. J. Brachman, H. J. Levesque, and R. Reiter, editors, *Principles of Knowledge Representation and Reasoning: Proceedings of the 1st International Conference*, pages 421–431, Toronto, ON, May 1989.

[29] J. G. Schmolze and W. S. Mark. The NIKL experience. *Computational Intelligence*, 6:48–69, 1991.

[30] N. K. Sondheimer and B. Nebel. A logical-form and knowledge-base design for natural language generation. In *Proceedings of the 5th National Conference of the American Association for Artificial Intelligence*, pages 612–618, Philadelphia, PA, Aug. 1986.

[31] W. Wahlster, E. Andre, S. Bandyopadhyay, W. Graf, and T. Rist. WIP: the coordinated generation of multimodal presentations from a common representation. In A. Ortony, J. Slack, and O. Stock, editors, *Computational Theories of Communication and their Applications*. Springer-Verlag, Berlin, Heidelberg, New York, 1991. To appear. Also available as DFKI Research Report RR-91-08.

[32] B. C. Williams. Interaction-based invention: Designing novel devices from first principles. In *Proceedings of the 8th National Conference of the American Association for Artificial Intelligence*, pages 349–356, Boston, MA, Aug. 1990. MIT Press.

A The Common Terminological Language CTL

The common semantic framework of terminological representation systems can be sketched as follows. There exists a number of atoms that belong to different syntactic categories. Usually, there are at least two disjoint categories, *roles* and *concepts*. Sometimes also *individuals* and *attributes* are part of the vocabulary. The meaning of these atoms is specified set-theoretically. A concept atom denotes some subset of the universe, a role atom denotes a two-place relation over the universe, an attribute atom denotes a partial function, and an individual atom denotes one element of the universe. A number of concept- and role-forming operators can be used to form new concept and role expressions where the meaning of these expressions is determined compositionally from its parts. Additionally, there are some operators to assign the meaning of an expression to an atom, to form so-called "terminological axioms". Finally, there may be some operators to specify restrictions on the interpretation of atoms, for instance, that two concept atoms are disjoint or that they cover the interpretation of another concept.

We assume four different alphabets of symbols, namely,

- the set of *individual symbols* $\mathbf{I}$, where i and j denote elements of $\mathbf{I}$;
- the set of *concept symbols* $\mathbf{A}$, where A and B are used to denote elements of $\mathbf{A}$;
- the set of *role symbols* $\mathbf{P}$, where P and Q denote elements of $\mathbf{P}$;
- the set of *attribute symbols* (or *feature symbols*) $\mathbf{p}$, where p and q denote elements of $\mathbf{p}$.

These symbols are interpreted in a set-theoretical way, i.e., we assume *interpretations* $\mathcal{I} = \langle \mathcal{D}, [\![\cdot]\!]^{\mathcal{I}} \rangle$, where $\mathcal{D}$ is an arbitrary set and $[\![\cdot]\!]^{\mathcal{I}}$ is a function

$$[\![\cdot]\!]^{\mathcal{I}}: \begin{cases} \mathbf{I} & \rightarrow & \mathcal{D} \\ \mathbf{A} & \rightarrow & 2^{\mathcal{D}} \\ \mathbf{P} & \rightarrow & 2^{\mathcal{D}\times\mathcal{D}} \\ \mathbf{p} & \rightarrow & 2^{\mathcal{D}\times\mathcal{D}} \end{cases}$$

such that there is at most one element $y \in \mathcal{D}$ for all $x \in \mathcal{D}$ with $(x, y) \in [\![p]\!]^{\mathcal{I}}$ for all $p \in \mathbf{p}$, and such that $[\![i]\!]^{\mathcal{I}} \neq [\![j]\!]^{\mathcal{I}}$, for all $i \neq j$. We will also use a functional notation for roles, i.e. $[\![R]\!]^{\mathcal{I}}(d) = \{e | (d, e) \in [\![R]\!]^{\mathcal{I}}\}$.

Using these sets of symbols, concept descriptions (denoted by C and D), role descriptions (R and S), role-chains ($\mathcal{RC}$ and $\mathcal{SC}$), attribute descriptions (r and s), and attribute chains ($\mathcal{r}$ and $\mathcal{s}$) can be constructed according to the abstract syntax rules shown in the tables below. In particular, the set of concept-, role-, and attribute-forming operators for constructing new concept and role expressions are defined in the Tables 1 and 2, respectively. The operators for specifying restrictions on the interpretation of atoms are summarized in Table 3. Finally, the set of terminological axioms is given in Table 4.

CONCEPT FORMING OPERATORS		
Abstract form:	Concrete form:	Interpretation:
$C, D \rightarrow A$	A	$[\![A]\!]^{\mathcal{I}}$
$\mid C \sqcap D$	(and C D)	$[\![C]\!]^{\mathcal{I}} \cap [\![D]\!]^{\mathcal{I}}$
$\mid C \sqcup D$	(or C D)	$[\![C]\!]^{\mathcal{I}} \cup [\![D]\!]^{\mathcal{I}}$
$\mid \neg C$	(not C)	$\mathcal{D} \setminus [\![C]\!]^{\mathcal{I}}$
$\mid \forall R\!:\!C$	(all R C)	$\{d \in \mathcal{D} \mid [\![R]\!]^{\mathcal{I}}(d) \subseteq [\![C]\!]^{\mathcal{I}}\}$
$\mid \exists R$	(some R)	$\{d \in \mathcal{D} \mid [\![R]\!]^{\mathcal{I}}(d) \neq \emptyset\}$
$\mid \geq nR$	(atleast n R)	$\{d \in \mathcal{D} \mid \lvert[\![R]\!]^{\mathcal{I}}(d)\rvert \geq n\}$
$\mid \leq nR$	(atmost n R)	$\{d \in \mathcal{D} \mid \lvert[\![R]\!]^{\mathcal{I}}(d)\rvert \leq n\}$
$\mid nR$	(exact n R)	$\{d \in \mathcal{D} \mid \lvert[\![R]\!]^{\mathcal{I}}(d)\rvert = n\}$
$\mid (0 \vee \geq n)R$	(optatleast n R)	$\{d \in \mathcal{D} \mid [\![R]\!]^{\mathcal{I}}(d) = \emptyset \vee \lvert[\![R]\!]^{\mathcal{I}}(d)\rvert \geq n\}$
$\mid \exists R\!:\!C$	(some R C)	$\{d \in \mathcal{D} \mid [\![R]\!]^{\mathcal{I}}(d) \cap [\![C]\!]^{\mathcal{I}} \neq \emptyset\}$
$\mid \geq nR\!:\!C$	(atleast n R C)	$\{d \in \mathcal{D} \mid \lvert[\![R]\!]^{\mathcal{I}}(d) \cap [\![C]\!]^{\mathcal{I}}\rvert \geq n\}$
$\mid \leq nR\!:\!C$	(atmost n R C)	$\{d \in \mathcal{D} \mid \lvert[\![R]\!]^{\mathcal{I}}(d) \cap [\![C]\!]^{\mathcal{I}}\rvert \leq n\}$
$\mid nR\!:\!C$	(exact n R C)	$\{d \in \mathcal{D} \mid \lvert[\![R]\!]^{\mathcal{I}}(d) \cap [\![C]\!]^{\mathcal{I}}\rvert = n\}$
$\mid \mathcal{RC} = \mathcal{SC}$	(eq $\mathcal{RC}$ $\mathcal{SC}$)	$\{d \in \mathcal{D} \mid [\![\mathcal{RC}]\!]^{\mathcal{I}}(d) = [\![\mathcal{SC}]\!]^{\mathcal{I}}(d)\}$
$\mid \mathcal{RC} \neq \mathcal{SC}$	(neq $\mathcal{RC}$ $\mathcal{SC}$)	$\{d \in \mathcal{D} \mid [\![\mathcal{RC}]\!]^{\mathcal{I}}(d) \neq [\![\mathcal{SC}]\!]^{\mathcal{I}}(d)\}$
$\mid \mathcal{RC} \subseteq \mathcal{SC}$	(subset $\mathcal{RC}$ $\mathcal{SC}$)	$\{d \in \mathcal{D} \mid [\![\mathcal{RC}]\!]^{\mathcal{I}}(d) \subseteq [\![\mathcal{SC}]\!]^{\mathcal{I}}(d)\}$
$\mid r : C$	(in r C)	$\{d \in \mathcal{D} \mid \emptyset \neq [\![r]\!]^{\mathcal{I}}(d) \subseteq [\![C]\!]^{\mathcal{I}}\}$
$\mid r : i$	(is r i)	$\{d \in \mathcal{D} \mid [\![i]\!]^{\mathcal{I}} \in [\![r]\!]^{\mathcal{I}}(d)\}$
$\mid rc = sc$	(eq rc sc)	$\{d \in \mathcal{D} \mid [\![rc]\!]^{\mathcal{I}}(d) = [\![sc]\!]^{\mathcal{I}}(d)\}$
$\mid rc \neq sc$	(neq rc sc)	$\{d \in \mathcal{D} \mid [\![rc]\!]^{\mathcal{I}}(d) \neq [\![sc]\!]^{\mathcal{I}}(d)\}$
$\mid rc \doteq sc$	(eq rc sc)	$\{d \in \mathcal{D} \mid [\![rc]\!]^{\mathcal{I}}(d) = [\![sc]\!]^{\mathcal{I}}(d) \neq \emptyset\}$
$\mid rc \not\doteq sc$	(neq rc sc)	$\{d \in \mathcal{D} \mid \emptyset \neq [\![rc]\!]^{\mathcal{I}}(d) \neq [\![sc]\!]^{\mathcal{I}}(d) \neq \emptyset\}$
$\mid \{i_1, i_2, \ldots, i_n\}$	(oneof $i_1 \ldots i_n$)	$\{[\![i_1]\!]^{\mathcal{I}}, [\![i_2]\!]^{\mathcal{I}}, \ldots, [\![i_n]\!]^{\mathcal{I}}\}$
$\mid$ datatype		
$\mid fn$	(apply fn)	

Table 1: CTL: Concept forming operators

ROLE AND ATTRIBUTE FORMING OPERATORS		
Abstract form:	Concrete form:	Interpretation:
$R, S \rightarrow P$	P	$[\![P]\!]^{\mathcal{I}}$
$\mid R \sqcap S$	(and R S)	$[\![R]\!]^{\mathcal{I}} \cap [\![S]\!]^{\mathcal{I}}$
$\mid R \sqcup S$	(or R S)	$[\![R]\!]^{\mathcal{I}} \cup [\![S]\!]^{\mathcal{I}}$
$\mid R^{-1}$	(inverse R)	$\{(d, d') \mid (d', d) \in [\![R]\!]^{\mathcal{I}}\}$
$\mid r^{-1}$	(inverse r)	$\{(d, d') \mid (d', d) \in [\![r]\!]^{\mathcal{I}}\}$
$\mid R\vert_C$	(restr R C)	$\{(d, d') \in [\![R]\!]^{\mathcal{I}} \mid d' \in [\![C]\!]^{\mathcal{I}}\}$
$\mid {}_C\vert R$	(domrestr R C)	$\{(d, d') \in [\![R]\!]^{\mathcal{I}} \mid d \in [\![C]\!]^{\mathcal{I}}\}$
$\mid C \times D$	(domainrange C D)	$[\![C]\!]^{\mathcal{I}} \times [\![D]\!]^{\mathcal{I}}$
$\mid R^{+}$	(trans R)	$([\![R]\!]^{\mathcal{I}})^{+}$
$\mid \mathcal{RC}$		
$\mid r$		
$\mathcal{RC} \rightarrow (R_1, \ldots, R_n)$	(compose $R_1 \ldots R_n$)	$[\![R_1]\!]^{\mathcal{I}} \circ \ldots \circ [\![R_n]\!]^{\mathcal{I}}$
$\mid \mathbf{1}$	self	$\{(d, d) \mid d \in \mathcal{D}\}$
$r \rightarrow p$	p	$[\![p]\!]^{\mathcal{I}}$
$\mid r\vert_C$	(restr r C)	$\{(d, d') \in [\![r]\!]^{\mathcal{I}} \mid d' \in [\![C]\!]^{\mathcal{I}}\}$
$\mid rc$		
$rc \rightarrow (r_1, \ldots, r_n)$	(compose $r_1 \ldots r_n$)	$[\![r_1]\!]^{\mathcal{I}} \circ \ldots \circ [\![r_n]\!]^{\mathcal{I}}$
$\mid \mathbf{1}$	self	$\{(d, d) \mid d \in \mathcal{D}\}$

Table 2: CTL: Role and attribute forming operators

ADDITIONAL RESTRICTIVE OPERATORS		
Abstract form:	Concrete form:	Interpretation:
$A_1 \| \ldots \| A_n$	(disjoint $A_1 \ldots A_n$)	$[\![A_i]\!]^{\mathcal{I}} \cap [\![A_j]\!]^{\mathcal{I}} = \emptyset, i \neq j$
$\{A_1, \ldots, A_n\} \bigcirc B$	(cover B $A_1 \ldots A_n$)	$\bigcup_{k=1}^{n} [\![A_k]\!]^{\mathcal{I}} \supseteq [\![B]\!]^{\mathcal{I}}$
$\{A_1, \ldots, A_n\} \textcircled{\|} B$	(disjcover B $A_1 \ldots A_n$)	$\bigcup_{k=1}^{n} [\![A_k]\!]^{\mathcal{I}} \supseteq [\![B]\!]^{\mathcal{I}}$, $[\![A_i]\!]^{\mathcal{I}} \cap [\![A_j]\!]^{\mathcal{I}} = \emptyset, i \neq j$
$\lvert A \rvert \leq 1$	(indiv A)	$\lvert [\![A]\!]^{\mathcal{I}} \rvert \leq 1$
$A \rightarrow B$	(implies A B)	$[\![A]\!]^{\mathcal{I}} \subseteq [\![B]\!]^{\mathcal{I}}$

Table 3: CTL: Additional restrictive operators

TERMINOLOGICAL AXIOMS		
Abstract form:	Concrete form:	Interpretation:
$A \doteq C$	(defconcept A C)	$[\![A]\!]^{\mathcal{I}} = [\![C]\!]^{\mathcal{I}}$
$A \sqsubseteq C$	(defprimconcept A $\{C\}$)	$[\![A]\!]^{\mathcal{I}} \subseteq [\![C]\!]^{\mathcal{I}}$
$P \doteq R$	(defrole P R)	$[\![P]\!]^{\mathcal{I}} = [\![R]\!]^{\mathcal{I}}$
$P \sqsubseteq R$	(defprimrole P R)	$[\![P]\!]^{\mathcal{I}} \subseteq [\![R]\!]^{\mathcal{I}}$
$P \sqsubseteq \top$	(defprimrole P)	$[\![P]\!]^{\mathcal{I}} \subseteq \mathcal{D} \times \mathcal{D}$
$p \doteq r$	(defattribute p r)	$[\![p]\!]^{\mathcal{I}} = [\![r]\!]^{\mathcal{I}}$
$p \sqsubseteq \top$	(defprimattribute p)	$[\![p]\!]^{\mathcal{I}} \in \mathcal{D} \times \mathcal{D}$

Table 4: CTL: Terminological axioms

B Expressiveness of the Systems

CONCEPT FORMING OPERATORS						
	System					
	BACK	CLASSIC	KRIS	LOOM	MESON	SB-ONE
A	Y	Y	Y	Y	Y	Y
$C \sqcap D$	Y	Y	Y	Y	Y	Y
$C \sqcup D$			Y	Y		
$\neg C$			Y			
$\forall R\colon C$	Y	Y	Y	Y	Y	Y
$\exists R$	I	I	Y	Y	I	I
$\geq nR$	Y	Y	Y	Y	Y	Y
$\leq nR$	Y	Y	Y	Y	Y	Y
nR	I	I	I	Y	Y	I
$(= 0 \vee \geq n)R$			I	I		Y
$\exists R\colon C$			Y	Y		
$\geq nR\colon C$				I		
$\leq nR\colon C$				I		
$nR\colon C$				I		
$\mathcal{RC} = \mathcal{SC}$				Y	Y	Y
$\mathcal{RC} \neq \mathcal{SC}$				Y		Y
$\mathcal{RC} \subseteq \mathcal{SC}$				Y	Y	Y
$r : C$		Y	Y	Y		
$r : i$		Y		Y		
$rc = sc$				Y		
$rc \neq sc$				Y		
$rc \stackrel{\downarrow}{=} sc$		Y	Y	Y	I	I
$rc \stackrel{\downarrow}{\neq} sc$			Y	Y		I
$\{i_1, i_2, \ldots, i_m\}$	Y	Y		Y		
datatype	Y	Y		Y	Y	
fn		Y		Y	Y	

Y – explicitly present
I – implicitly present

Table 5: Expressiveness: Concept forming operators

This section examines the expressiveness of the systems. In particular, Table 5 (Table 6) determines for every concept forming operator (role and attribute forming operator) its explicit or implicit existence in a system. Table 7 summarizes the available additional restrictive operators. Finally, Table 8 lists the terminological axioms that are defined for a particular system.

ROLE AND ATTRIBUTE FORMING OPERATORS						
	System					
	BACK	CLASSIC	KRIS	LOOM	MESON	SB-ONE
P	Y	Y	Y	Y	Y	Y
$R \sqcap S$	Y		Y	Y		
$R \sqcup S$						
R^{-1}				Y	Y	
r^{-1}				Y		
$R\|_C$	I			Y	Y	Y
$_C\|R$	I			Y		Y
$C \times D$	Y			I		Y
R^+						
$R\mathbb{C}$				Y	Y	Y
r		Y	Y	Y		
$(R_1, \ldots, R_n)$				Y	Y	Y
$\mathbf{1}$				Y		
p		Y	Y	Y		
$r\|_C$				Y		
$r\mathbb{c}$		I	Y	Y		
$(r_1, \ldots, r_n)$		Y	Y	Y		
$\mathbf{1}$				Y		

Y – explicitly present
I – implicitly present

Table 6: Expressiveness: Role and attribute forming operators

ADDITIONAL RESTRICTIVE OPERATORS						
	System					
	BACK	CLASSIC	KRIS	LOOM	MESON	SB-ONE
$A_1 \| \ldots \| A_n$	Y	Y	I	I	Y	Y
$\{A_1, \ldots, A_n\} \bigcirc B$				I		Y
$\{A_1, \ldots, A_n\} \textcircled{\|} B$				Y		I
$\|A\| \leq 1$	Y				Y	Y
$A \rightarrow B$		Y		Y	Y	

Y – explicitly present
I – implicitly present

Table 7: Expressiveness: Additional restrictive operators

TERMINOLOGICAL AXIOMS						
	System					
	BACK	CLASSIC	KRIS	LOOM	MESON	SB-ONE
(defconcept A C)	Y	Y	Y	Y	Y	Y
(defprimconcept A $\{C\}$)	Y	Y	Y	Y	Y	Y
(defrole P R)			Y	Y	Y†	Y
(defprimrole P R)	Y		Y	Y	Y†	Y
(defprimrole P)	Y	Y	Y	Y		Y
(defattribute p r)			Y	Y		
(defprimattribute p)		Y	Y	Y		

† – only inside concept definitions

Table 8: Expressiveness: Terminological axioms

C Problematic Cases

In the following, we use the abstract syntax to specify the tests we designed. Note that at some points the systems do not support the original language constructs, so we had to use a different formulation to check the system behavior. The alternative formulations are not always semantically equivalent. However, in the context where we used them they led to identical results. For instance, when concept disjointness $A\|B$ was not available, we used disjoint number restrictions to enforce the disjointness of the concepts as shown in Table 9. The results of the tests of problematic cases described below are given in Table 10.

used construct	alternative construct
$C\|\|D$	$C \doteq \geq 2R,\ D \doteq \leq 1R$
$\{C, D\} \bigcirc A$	$A \doteq C \sqcup D$
$C \doteq \geq nR\colon D$	$R_1 \sqsubseteq R,\ C \doteq (\geq nR_1) \sqcap (\forall R_1\colon D)$
$C \doteq \leq nR\colon D$	$R_1 \sqsubseteq R,\ C \doteq (\leq nR_1) \sqcap (\forall R_1\colon D)$
$C \doteq \forall R\colon (D \sqcap E)$	$C_1 \doteq \forall R\colon D,\ C_2 \doteq \forall R\colon E,\ C \doteq C_1 \sqcap C_2$

Table 9: Problematic Cases: Alternative constructs

The tests are as follows:

1. How does the system handle *syntactically incorrect input*? Using a LISP-like notation, we checked what happened in the following cases:

   (a) `(dfconcept a)`

   (b) `(defconcept a b c)`

   (c) `(defrole a (domain))`

   (d) `(defconcept a (domain b))`

2. How does the system react to *inconsistent* concepts, i.e., to concepts with a necessarily empty interpretation? Although this is completely legal in the semantic framework sketched above, and does not lead to the inconsistency of the knowledge base as a whole, the system designer may choose to raise an error or output a warning message. Two cases have to be distinguished here. First, a *concept subexpression* may be inconsistent, which does not lead to the inconsistency of the embedding expression. Second, a concept definition may result in an inconsistent atomic concept, which is rather useless—and most probably an error. We used the following expressions to check out the system behavior:

   (a) $E \doteq (\geq 2R) \sqcap (\leq 1R)$

   (b) $C\|D,\ E \doteq (\forall R\colon (C \sqcap D))$

3. How are *semantically equivalent concept* definitions handled? Again, this is perfectly legal in the semantic framework, but the system designer may have decided to issue a warning because such equivalent definitions seem to be useless.

   (a) $C \doteq \leq 1R,\ D \doteq C \sqcap (\geq 1R),\ E \doteq 1R$

   (b) $C \doteq \forall R\colon D,\ E \doteq \forall R\colon D$

4. If a concept or role *symbol appears more than once* on the left hand side of a definition symbol, there are two options to handle this case. First, one may decide to prohibit this and issue an error message. Secondly, it is possible to interpret the second definition as a revision, i.e., the first definition is discarded and the second definition is used as the actual definition.

   (a) $C \doteq \geq 1R,\ D \doteq \leq 1R,\ E \doteq C \sqcap D,\ C \doteq \geq 2R$
   $\vdash E \approx \bot$

   (b) $B \| C,\ R \sqsubseteq \top \times D,\ E \doteq (\geq 1R) \sqcap (\forall R\colon B),\ R \sqsubseteq \top \times C$
   $\vdash E \approx \bot$

5. What kind of *general constraints* are enforced? Some systems require that no concept atom is defined to be equivalent to the most general concept. Others enforce the restriction that if two concepts are declared to be disjoint, then they have to be *primitive concepts*, i.e., to specify only necessary but no sufficient conditions.

   (a) $C \doteq \top$

   (b) $F \doteq H,\ G \doteq I,\ F \| G$

6. Does the system accept *forward references* and the use of *previously undefined* concepts and roles? The semantic framework does not say anything about the linear order of concept and role definitions. Thus, the system designer has to choose whether all role and concept atoms have to be defined before they are used or whether forward definitions and "definitions by use" are permitted.

   (a) $D \doteq \geq 2R,\ C \doteq D \sqcap E,\ E \doteq \leq 1R$
   $\vdash \mathrm{C} \preceq \bot$

   (b) $C \| D,\ A \doteq \exists R\colon C,\ R \sqsubseteq B \times D$
   $\vdash \mathrm{A} \preceq \bot$

   (c) $C \doteq \exists R,\ S \sqsubseteq R$

7. If forward references are permitted, then it is possible to make circularly refering definitions. How does the system deal with them? In the semantic framework such *cycles* are no problem. It is not a trivial task to provide a reasonable semantics and inference algorithms for this case, however.

(a) $C \doteq \forall R\colon C$

(b) $C \doteq \forall R\colon C$, $D \doteq C \sqcap (\forall R\colon D)$
$\vdash \mathrm{C} \preceq \mathrm{D}$

(c) $C \doteq \forall R\colon D$, $D \doteq \forall R\colon C$
$\vdash C \approx D$

(d) $C \doteq \exists R\colon D$, $D \doteq \exists R\colon C$
$\vdash C \approx D$
(result depends on underlying semantics)

8. How does the system fix the *syntactic category* of a symbol? It is possible to use the same symbol as a role and as a concept atom, disambiguating by context—if the concrete syntax supports that. Does the system permit such overloading of symbols?

   (a) $C \doteq \exists R\colon D$, $E \doteq \exists C$

The Problematic Cases							
Result of Test ...		System					
		BACK	CLASSIC	KRIS	LOOM	MESON	SB-ONE
1	(a)	LError	CorrErr	CorrErr	CorrErr	CorrErr	LError
	(b)	LError	CorrErr	CorrErr	CorrErr	CorrErr	CorrErr
	(c)	LError	CorrErr	CorrErr	CorrErr	CorrErr	CorrErr
	(d)	—	CorrErr	CorrErr	CorrErr	CorrErr	CorrErr
2	(a)	Correct	Correct	Correct	Correct	CorrErr	CorrErr
	(b)	Correct	Correct	Correct	Correct	CorrErr	CorrErr
3	(a)	Correct	Correct	Correct	Correct	CorrErr	Correct
	(b)	Correct	Correct	Correct	Correct	CorrErr	Correct
4	(a)	Correct	CorrErr	Incorr	Correct	CorrErr	—
	(b)	Incorr	CorrErr	Incorr	—	—	—
5	(a)	Correct	CorrWarn	CorrWarn	CorrWarn	CorrErr	CorrWarn
	(b)	CorrErr	—	—	—	CorrErr	Correct
6	(a)	Correct	CorrErr	Correct	Correct	CorrErr	CorrErr
	(b)	Incorr	CorrErr	Correct	Correct	—	CorrErr
	(c)	Correct	CorrErr	Correct	Correct	Correct	CorrErr
7	(a)	CorrErr	CorrErr	CorrErr	(Correct)	(Correct)	(Correct)
	(b)	CorrErr	CorrErr	CorrErr	(Correct)	(Correct)	(Correct)
	(c)	CorrErr	CorrErr	CorrErr	(Correct)	CorrErr	(Correct)
	(d)	CorrErr	CorrErr	CorrErr	(Correct)	CorrErr	(Correct)
8	(a)	CorrErr	CorrErr	Incorr	Correct	—	Correct

LError	error message of underlying language
CorrErr	case is error in system interpretation and an error message is issued
CorrWarn	case is no error in system interpretation but a warning is issued
Correct	case is no error in system interpretation and handled correctly without warning
(Correct)	system accepts input but does not classify these concepts according to one of the possible semantics
Incorr	case is not handled correctly

Table 10: Problematic Cases: Test results

D TBox Inferences

In order to get a feeling "how complete" the systems are, we designed a number of small test cases with more or less obvious conclusions. The tests are by no means exhaustive or systematic, but some of them were designed with a slightly malicious attitude. As in Appendix C, we sometimes used reformulation of the tests given below by using the correspondences of Table 9.

It turned out that some systems failed to draw the right conclusion even in cases where they are supposed to be complete. Since these results have been given to the system designers, these "bugs" are probably not present in more recent system versions. Nevertheless, these results demonstrate that it would be profitable to have an exhaustive and systematic test suite available on which all systems could be tested.

1. One basic inference is the the detection of inconsistencies caused by disjoint concepts. We tested this for direct conjunctions (a), conjunctions appearing as value restrictions (b), and conjunctions of sub-concepts of two disjoint concepts (c). The case (d) may look a bit pathological, however, it is well-defined and easy to handle. Nevertheless, most systems failed on this example.

   (a) $C \| D,\ E \doteq C \sqcap D$
   $\vdash \mathrm{E} \preceq \bot$

   (b) $C \| D,\ E \doteq \forall R\colon (C \sqcap D) \sqcap \exists R$
   $\vdash \mathrm{E} \preceq \bot$

   (c) $C \| D,\ E \sqsubseteq C,\ F \sqsubseteq D$
   $\vdash E \| F$

   (d) $C \sqsubseteq D,\ C \| D$
   $\vdash \mathrm{C} \preceq \bot$

2. Another basic inference is to detect inconsistencies when the minimum and maximum restrictions ($\geq nR$ and $\leq nR$) on the same role are incompatible (a). A more complicated instance is case (b) where the disjointness of the concepts used in the existential quantification ($\exists R\colon C$) leads to the conclusion that there must be at least two role fillers. In fact, most systems are incomplete in this aspect that involves reasoning about the number of role fillers in the general case (see also test 3).

   (a) $E \doteq (\geq 2R) \sqcap (\leq 1R)$
   $\vdash \mathrm{E} \preceq \bot$

   (b) $C \| D,\ E \doteq (\leq 1R) \sqcap (\exists R\colon C) \sqcap (\exists R\colon D)$
   $\vdash \mathrm{E} \preceq \bot$

3. If a language contains minimum and maximum restrictions and subroles (i.e., the possibility of expressing role conjunctions) or qualified number restrictions

($\geq nR\colon C$ and $\leq nR\colon C$), then the reasoning about the number of potential role fillers can be become quite complicated. Case (a) is a comparably easy case resembling test 2 (b). Cases (b)–(e) are more complicated, but all inferences are based on the fact that because of disjointness and/or covering of role-filler concepts by another concept additional minimum or maximum restrictions for subroles (or qualified existential restrictions) could be derived.

(a) $C\|D,\ E \doteq (\geq 1R) \sqcap (\exists R\colon C) \sqcap (\exists R\colon D)$
$\vdash E \preceq (\geq 2R)$

(b) $C\|D,\ E \doteq (\leq 2R) \sqcap (\exists R\colon C) \sqcap (\exists R\colon D)$
$\vdash E \preceq (\leq 1R\colon C) \sqcap (\leq 1R\colon D)$

(c) $\{C, D\} \textcircled{\scriptsize ||} A,\ R \sqsubseteq T|_C,\ S \sqsubseteq T|_D,$
$E \doteq (\forall T\colon A) \sqcap (\geq 3T) \sqcap (\leq 1R) \sqcap (\leq 1S)$
$\vdash \mathrm{E} \preceq \bot$

(d) $\{C, D\} \bigcirc A,\ E \doteq (\forall R\colon A) \sqcap (\geq 3R) \sqcap (\leq 1R\colon C)$
$\vdash E \preceq (\geq 2R\colon D)$

(e) $R_1, R_2, R_3 \sqsubseteq R,\ T_1, T_2, T_3 \sqsubseteq T,\ C\|D\|E,$
$F_1 \doteq \exists R_1\colon ((\leq 2T) \sqcap (\exists T_1\colon C)))$
$F_2 \doteq \exists R_2\colon ((\leq 2T) \sqcap (\exists T_2\colon D)))$
$F_3 \doteq \exists R_3\colon ((\leq 2T) \sqcap (\exists T_3\colon E)))$
$F \doteq F_1 \sqcap F_2 \sqcap F_3$
$\vdash F \preceq (\geq 2R)$

4. The role-forming operator that restricts the range of a role ($R|_C$) leads to intractability in the general case [17]. In fact, this operator can be used to model "disjunctive reasoning." Case (a) is an example where *modus ponens* like reasoning is necessary. Case (b) shows an example where reasoning by case is required.

(a) $C \doteq (\forall R\colon D) \sqcap (\forall R|_D\colon E)$
$\vdash \mathrm{C} \preceq (\forall R\colon E)$

(b) $C\|D,\ E \doteq (\forall R|_{(\geq 2S)}\colon C) \sqcap \forall R\colon D$
$\vdash E \preceq \forall R\colon (\leq 1S)$

5. Role-value maps ($RC = SC$) are known to cause undecidability of subsumption [28]. However, even if role-value maps are handled in an incomplete manner, there are still some inferences that can be easily computed, for instance, inconsistencies caused by conflicting value, minimum or maximum restrictions, as in the cases (a)–(b), or caused by conflicting role-value map specifications, as in case (c).

(a) $E \doteq (1R) \sqcap (2S) \sqcap (R = S)$
$\vdash \mathrm{E} \preceq \bot$

(b) $C \| D,\ E \doteq (\forall R\colon C) \sqcap (\forall S\colon D) \sqcap (R = S) \sqcap (\exists R)$
$\vdash \mathrm{E} \preceq \bot$

(c) $C \doteq (\leq 1R) \sqcap (R = S),\ D \doteq (\leq 1R) \sqcap (R \neq S)$
$\vdash C \| D,$

6. While equality reasoning over chains of roles is undecidable, equality reasoning over chains of attributes (i.e., single-valued roles) is quite easy [2] and can be easily added to terminological languages [15]. Interestingly, the two systems that claimed to be complete in this aspect, namely, CLASSIC and KRIS failed on at least one of the tests given below.

(a) $E \doteq (rst \stackrel{\downarrow}{=} uv) \sqcap (rstx \stackrel{\downarrow}{=} vu)$
$\vdash E \preceq rstx \stackrel{\downarrow}{=} uvx,\ E \preceq uvx \stackrel{\downarrow}{=} vu$

(b) $F \doteq (rst \stackrel{\downarrow}{=} uvw) \sqcap (rs \stackrel{\downarrow}{=} u)$
$\vdash F \preceq rsvw \stackrel{\downarrow}{=} ut$

7. Finally, we tested the role-forming operator *inverse* (R^{-1}).

(a) $E \doteq \forall R\colon (\forall R^{-1}\colon A) \sqcap \exists R$
$\vdash E \preceq A$

The results of the tests are displayed in Table 11. Most of them simply confirm the formal or informal specification of the system.

BACK: The negative results for the tests 2 (b) and 2 (a)–(e) are not surprising since BACK does not compute the cardinality of role-filler sets of sub- and superroles [27]. The negative result for test 1 is surprising, however.

CLASSIC: The interesting point is that CLASSIC fails on case 6 (a), although CLASSIC is supposed to be complete in this aspect.

KRIS: Even more surprising to us was the fact that the tested version of KRIS was incomplete for many examples. However, we tested one of the first prototypical versions. Further, we used one particular subsumption algorithm, namely, the $\mathcal{ALCFN}$-subsumption algorithm. When using the more powerful but slower $\mathcal{ALCFNR}$-subsumption algorithm, these errors did not occur.

LOOM: The incompleteness of LOOM was no big surprise to us since this system is explicitly described as supporting a very expressive terminological language in an incomplete manner. Some of the tests may actually lead to a positive result if "ABox"-reasoning is used. Since we tested only the terminological part of the system, this was not taken into account.

TBox Inferences							
Result of Test ...		System					
		BACK	CLASSIC	KRIS	LOOM	MESON	SB-ONE
1	(a)	Y	Y	Y	Y	Y	Y
	(b)	Y	Y	N	Y	Y	Y
	(c)	Y	Y	Y	Y	Y	Y
	(d)	N	—	—	—	N	N
2	(a)	Y	Y	Y	Y	Y	Y
	(b)	N	—	Y	N	Y	Y
3	(a)	N	—	Y	N	Y	Y
	(b)	N	—	—	N	Y	N
	(c)	N	—	—	N	—	Y
	(d)	N	—	—	N	—	N
	(e)	N	—	N/Y	N	—	—
4	(a)	—	—	—	N	N	—
	(b)	—	—	—	N	—	—
5	(a)	—	—	—	Y	N	Y
	(b)	—	—	N/Y	Y	—	Y
	(c)	—	—	N/Y	N	—	N
6	(a)	—	N	N/Y	N	—	—
	(b)	—	Y	N/Y	N	—	—
7	(a)	—	—	—	N	Y	—

Y inference drawn
N inference not drawn
N/Y result depends on used subsumption algorithm

Table 11: TBox Inferences: Test results

MESON: The incompleteness in case 4 (a) is no surprise. For the other two cases, a similar remark as above applies. The conclusions are drawn if "ABox"-reasoning is employed.

SB-ONE: The interesting point about SB-ONE is that it tries to account for reasoning with the cardinality of role-filler sets for sub- and superroles, in an incomplete manner, though.

E Hard Cases

Below, four types of "hard cases" are decribed. The results of the individual tests are given in Table 12:

1. When computing the subsumption relation between C_n and D_n, some systems may expand the definitions. Each concept has two roles with the same value restrictions which will be replaced by their definition. This leads to an exponential increase of the definitions.

$$\begin{aligned} C_0 &\sqsubseteq \top \\ C_1 &\doteq (\forall R_1 : C_0) \sqcap (\forall R_2 : C_0) \\ C_2 &\doteq (\forall R_1 : C_1) \sqcap (\forall R_2 : C_1) \\ &\vdots \\ C_n &\doteq (\forall R_1 : C_{n-1}) \sqcap (\forall R_2 : C_{n-1}) \end{aligned}$$

   In analogy, the concepts D_0, D_1, ..., D_n will be defined.

   Test: $C_n \preceq D_n$ for $n = 4, 8, 12$

2. The following test examines the run time when computing value restrictions that are concept conjunctions. If the value restrictions are expanded straightforwardly without checking whether they are identical, we get the exponential increase of the definitions as in the case above.

$$\begin{aligned} C_0 &\sqsubseteq \top \\ C_1 &\sqsubseteq \top \\ C_2 &\doteq \forall R_1 : (C_0 \sqcap C_1) \\ C_3 &\doteq \forall R_2 : (C_0 \sqcap C_1) \\ &\vdots \\ C_{2n} &\doteq \forall R_1 : (C_{2n-2} \sqcap C_{2n-1}) \\ C_{2n+1} &\doteq \forall R_2 : (C_{2n-2} \sqcap C_{2n-1}) \end{aligned}$$

   In analogy, the concepts D_0, D_1, ..., D_{2n+1} will be defined.

   Test: $C_{2n+1} \preceq D_{2n+1}$ for $n = 4, 8, 12$

3. The next test is an example for the fact that in (nearly) every system you can get exponential answer times. We model a sequence of concepts in a way such that even if the best conceivable algorithm is used (resembling an algorithm to decide equivalence for non-deterministic finite state automata) exponential time is necessary [20].

$$C_{2n} \sqsubseteq \top$$
for $n < i \leq 2n-1$:
$$C_i \doteq (\forall R_1 \colon C_{i+1}) \sqcap (\forall R_2 \colon C_{i+1})$$
for $0 \leq i \leq n$:
$$C_i \doteq (\forall R_1 \colon C_{i+1}) \sqcap (\forall R_2 \colon (C_{i+1} \sqcap C_{2i}))$$

In order to trigger the creation of the new value restrictions (not all systems do this automatically without need) the concepts D_0 to D_{2n} for $n = 4, 8, 12$ are defined analogously and the subsumption relation between C_0 and D_0 is computed:

Test: $C_0 \preceq D_0$ for $n = 4, 8, 12$

4. The last test was inspired from the algorithms in [13]. The restrictions in the "$\forall R$" parts have to be propagated to the two "$\exists R$" parts (because they should hold for every R) and this leads to an exponential increase of the definition.

$E \doteq \forall R \colon A$
$F \doteq \forall R \colon B$
$G \doteq \forall R \colon (A \sqcap B)$
C $\doteq$

$$n\text{-times} \left\{ \begin{array}{l} (\exists R \colon E) \sqcap (\exists R \colon G) \sqcap \\ \forall R \colon (G \sqcap (\exists R \colon E) \sqcap (\exists R \colon G) \sqcap \\ \quad \forall R \colon (G \sqcap (\exists R \colon E) \sqcap (\exists R \colon G) \sqcap \\ \quad\quad \forall R \colon (G \sqcap (\exists R \colon E) \sqcap \ldots \\ \quad\quad\quad \vdots \\ \quad\quad\quad \forall R \colon (G \sqcap (\exists R \colon E) \sqcap (\exists R \colon G)) \ldots) \end{array} \right.$$

In analogy, concept D is defined on the basis of F (instead of E) and the subsumption relations between C and D are computed for $n = 4, 8, 12$:

Test: $C \preceq D$ for $n = 4, 8, 12$

The Hard Cases							
Result (sec) of Test ...		System					
		BACK	CLASSIC	KRIS	LOOM	MESON	SB-ONE
1	n=4	1	2	3	1	1	11
	n=8	1	4	77	3	2	33
	n=12	2	5	2680	5	6	56
2	n=4	7	3	82	7	3	99
	n=8	32	11	1867	22	23	859
	n=12	75	16	—†	39	84	3263
3	n=4	25	4	459	28	29	372
	n=6	352	40	18230	155	5099	1836
	n=8	6035	706	—†	666	—†	9500
4	n=4	—‡	—	4	4	—	—
	n=8	—	—	49	8	—	—
	n=12	—	—	745	13	—	—

† test has been aborted

‡ system died on this example

Table 12: Hard Cases: Test results

F Real Knowledge Bases

Below we give a brief description of the "real" knowledge bases used as one part of our tests. Table 13 characterizes the structure of the original KBs by means of the number of defined and primitive concepts and roles, respectively. As mentioned above, by automatically translating and adapting the KBs to each particular system, some artificial concepts have been introduced, the cardinality is also shown in Table 13. The exact number of auxiliary concepts may differ from system to system, though.

Table 14 summarizes the results in terms of total run time needed to sequentially load and classify a KB. The third column shows the time needed for loading and *simultaneously* classifying the KB. Only the systems LOOM and MESON allow both modes of loading. In the case of the BACK system we gave up on producing a version of a "Wisber" KB both similar to the original one and acceptable to BACK.

CKB (Conceptual Knowledge Base): Contains knowledge about tax regulations and is used in the Natural Language project XTRA at the University of Saarbrücken.

Companies: Contains knowledge about company structures and is used at the Technical University Berlin in the framework of the ESPRIT project ADKMS.

FSS (Functional Semantic Structures): Contains knowledge about speech acts and is used in the Natural Language project XTRA at the University of Saarbrücken.

Espresso: Contains knowledge about Espresso machines and their structure. It is used in the WIP-Project of DFKI in the framework of multimodal presentation of information.

Wisber: Contains knowledge about different forms of investments and was used in the natural language dialog project WISBER at the University of Hamburg.

Wines simple kosher: Contains knowledge about wines, wineries, and meal-courses. It is used as sample KB of the CLASSIC system.[5]

[5] A lot of individuals have been transformed to general concepts because in our tests we only considered terminological knowledge but did not want to cut all the nice information about different wineries and wines.

THE STRUCTURE OF THE REAL KBs							
Name	Original Language	defined	primitive	artificial	$\sum$	defined	primitive
		concepts				roles	
CKB	SB-ONE	23	57	58	138	2	46
Companies	BACK	70	45	81	196	1	39
FSS	SB-ONE	34	98	75	207	0	47
Espresso	SB-ONE	0	145	79	224	11	41
Wisber	TURQ	50	81	152	283	6	18
Wines	CLASSIC	50	148	237	435	0	10

Table 13: Real Knowledge Bases: Structural description

TESTING THE REAL KBs				
KB	System	Total run time (sec) needed to ... load	classify	ld+cl
CKB	BACK	—	—	9
	CLASSIC	—	—	9
	KRIS	4	580	—
	LOOM	14	13	27
	MESON	22	138	43
	SB-ONE	60	65	—
Companies	BACK	—	—	41
	CLASSIC	—	—	14
	KRIS	5	2607	—
	LOOM	20	21	45
	MESON	67	98	97
	SB-ONE	583	802	—
FSS	BACK	—	—	128
	CLASSIC	—	—	20
	KRIS	5	8170	—
	LOOM	20	26	75
	MESON	274	447	255
	SB-ONE	330	1176	—
Espresso	BACK	—	—	47
	CLASSIC	—	—	17
	KRIS	7	1944	—
	LOOM	24	29	55
	MESON	41	42	58
	SB-ONE	87	211	—
Wisber	BACK	—	—	—†
	CLASSIC	—	—	22
	KRIS	6	6223	—
	LOOM	24	28	61
	MESON	151	142	213
	SB-ONE	1510	3417	—
Wines	BACK	—	—	486
	CLASSIC	—	—	56
	KRIS	10	45327	—
	LOOM	36	47	92
	MESON	857	696	1321
	SB-ONE	1377	4500	—

† we gave up on producing a version of a "Wisber" KB acceptable to BACK

Table 14: Real Knowledge Bases: Test results

G Random Knowledge Bases

In order to eliminate the inaccuracy of measurement introduced by the translation process, to get an idea how the runtime varies with the number of concepts, and to test the systems on larger knowledge bases, a number of terminological knowledge bases were randomly generated using only the intersection of all terminological languages used in the systems (i.e., only conjunction, value restrictions, and number restrictions). The structure of these generated knowledge bases resembles some of the aspects of the real knowledge bases we used. We do not claim, however, that the generated knowledge bases are realistic in all aspects.

The generated knowledge bases have the following properties:

- 80% of the concepts are "primitive" (i.e., introduced by $\sqsubseteq$).
- There are exactly 10 different roles.
- Each concept definition is a conjunction containing
  - one or two concept symbols (explicit super-concepts),
  - zero or one minimum restrictions,
  - zero or one maximum restrictions,
  - and zero, one, or two value restrictions,

  where the number of constructs from one category and the roles and concepts are randomly assigned with a uniform distribution. Further, the concepts are constructed in a way such that no concept is inconsistent (i.e., no minimum restriction is larger than any maximum restriction).

In order to avoid definitional cycles, the concepts are partitioned into *layers*, where the ith layer has 3^i concepts. When assigning explicit super-concepts or value-restriction concepts to the concept definition of a concept from level i, only concepts from level 0 to $i-1$ are considered.

Comparing the randomly generated knowledge bases with real knowledge bases, one notes that the number of roles might not be realistic. Further, the randomly generated knowledge bases tend to have a concept hierarchy that is less tree-like than real knowledge bases. Nevertheless, the runtime performance on the generated knowledge bases is comparable with the runtime performance on real knowledge bases.

Loading and Classifying a Random KB						
Average time (sec) for ... concepts	System					
	BACK	CLASSIC	KRIS	LOOM	MESON	SB-ONE
15	1	1	5	3	3	68
30	2	2	25	6	8	188
45	4	3	61	9	17	370
60	10	5	131	13	46	
75	12	5		19	46	
90	15	5		18	57	
105	21	8		19	87	
120	31	9		28	142	
135	38	11		32	190	
150	50	11		37	247	
200	91	13		46	550	
400	441	42		110	3500	
600	1519	65		251		
800	3490	134		354		
1000		157		480		
1200		180		643		
1400		260		853		
1600		359		1324		
1800		424		1520		
2000		493		1833		
2500		1441		3106		
3000		2704		3915		
4000		8195		5527		
5000				11019		

Table 15: Random knowledge bases: Test results

Deutsches
Forschungszentrum
für Künstliche
Intelligenz GmbH

DFKI
-Bibliothek-
PF 2080
D-6750 Kaiserslautern
FRG

DFKI Publikationen

Die folgenden DFKI Veröffentlichungen sowie die aktuelle Liste von allen bisher erschienenen Publikationen können von der oben angegebenen Adresse bezogen werden.
Die Berichte werden, wenn nicht anders gekennzeichnet, kostenlos abgegeben.

DFKI Publications

The following DFKI publications or the list of all publisched papers so far can be ordered from the above address.
The reports are distributed free of charge except if otherwise indicated.

DFKI Research Reports

RR-91-08
Wolfgang Wahlster, Elisabeth André, Som Bandyopadhyay, Winfried Graf, Thomas Rist: WIP: The Coordinated Generation of Multimodal Presentations from a Common Representation
23 pages

RR-91-09
Hans-Jürgen Bürckert, Jürgen Müller, Achim Schupeta: RATMAN and its Relation to Other Multi-Agent Testbeds
31 pages

RR-91-10
Franz Baader, Philipp Hanschke: A Scheme for Integrating Concrete Domains into Concept Languages
31 pages

RR-91-11
Bernhard Nebel: Belief Revision and Default Reasoning: Syntax-Based Approaches
37 pages

RR-91-12
J.Mark Gawron, John Nerbonne, Stanley Peters: The Absorption Principle and E-Type Anaphora
33 pages

RR-91-13
Gert Smolka: Residuation and Guarded Rules for Constraint Logic Programming
17 pages

RR-91-14
Peter Breuer, Jürgen Müller: A Two Level Representation for Spatial Relations, Part I
27 pages

RR-91-15
Bernhard Nebel, Gert Smolka: Attributive Description Formalisms ... and the Rest of the World
20 pages

RR-91-16
Stephan Busemann: Using Pattern-Action Rules for the Generation of GPSG Structures from Separate Semantic Representations
18 pages

RR-91-17
Andreas Dengel, Nelson M. Mattos: The Use of Abstraction Concepts for Representing and Structuring Documents
17 pages

RR-91-18
John Nerbonne, Klaus Netter, Abdel Kader Diagne, Ludwig Dickmann, Judith Klein: A Diagnostic Tool for German Syntax
20 pages

RR-91-19
Munindar P. Singh: On the Commitments and Precommitments of Limited Agents
15 pages

RR-91-20
Christoph Klauck, Ansgar Bernardi, Ralf Legleitner FEAT-Rep: Representing Features in CAD/CAM
48 pages

RR-91-21
Klaus Netter: Clause Union and Verb Raising Phenomena in German
38 pages

RR-91-22
Andreas Dengel: Self-Adapting Structuring and Representation of Space
27 pages

RR-91-23
Michael Richter, Ansgar Bernardi, Christoph Klauck, Ralf Legleitner: Akquisition und Repräsentation von technischem Wissen für Planungsaufgaben im Bereich der Fertigungstechnik
24 Seiten

RR-91-24
Jochen Heinsohn: A Hybrid Approach for Modeling Uncertainty in Terminological Logics
22 pages

RR-91-25
Karin Harbusch, Wolfgang Finkler, Anne Schauder: Incremental Syntax Generation with Tree Adjoining Grammars
16 pages

RR-91-26
M. Bauer, S. Biundo, D. Dengler, M. Hecking, J. Koehler, G. Merziger: Integrated Plan Generation and Recognition - A Logic-Based Approach -
17 pages

RR-91-27
A. Bernardi, H. Boley, Ph. Hanschke, K. Hinkelmann, Ch. Klauck, O. Kühn, R. Legleitner, M. Meyer, M. M. Richter, F. Schmalhofer, G. Schmidt, W. Sommer: ARC-TEC: Acquisition, Representation and Compilation of Technical Knowledge
18 pages

RR-91-28
Rolf Backofen, Harald Trost, Hans Uszkoreit: Linking Typed Feature Formalisms and Terminological Knowledge Representation Languages in Natural Language Front-Ends
11 pages

RR-91-29
Hans Uszkoreit: Strategies for Adding Control Information to Declarative Grammars
17 pages

RR-91-30
Dan Flickinger, John Nerbonne: Inheritance and Complementation: A Case Study of *Easy* Adjectives and Related Nouns
39 pages

RR-91-31
H.-U. Krieger, J. Nerbonne: Feature-Based Inheritance Networks for Computational Lexicons
11 pages

RR-91-32
Rolf Backofen, Lutz Euler, Günther Görz: Towards the Integration of Functions, Relations and Types in an AI Programming Language
14 pages

RR-91-33
Franz Baader, Klaus Schulz: Unification in the Union of Disjoint Equational Theories: Combining Decision Procedures
33 pages

RR-91-34
Bernhard Nebel, Christer Bäckström: On the Computational Complexity of Temporal Projection and some related Problems
35 pages

RR-91-35
Winfried Graf, Wolfgang Maaß: Constraint-basierte Verarbeitung graphischen Wissens
14 Seiten

RR-92-01
Werner Nutt: Unification in Monoidal Theories is Solving Linear Equations over Semirings
57 pages

RR-92-02
Andreas Dengel, Rainer Bleisinger, Rainer Hoch, Frank Hönes, Frank Fein, Michael Malburg: Π_{ODA}: The Paper Interface to ODA
53 pages

RR-92-03
Harold Boley: Extended Logic-plus-Functional Programming
28 pages

RR-92-04
John Nerbonne: Feature-Based Lexicons: An Example and a Comparison to DATR
15 pages

RR-92-05
Ansgar Bernardi, Christoph Klauck, Ralf Legleitner, Michael Schulte, Rainer Stark: Feature based Integration of CAD and CAPP
19 pages

RR-92-06
Achim Schupetea: Main Topics od DAI: A Review
38 pages

RR-92-07
Michael Beetz: Decision-theoretic Transformational Planning
22 pages

RR-92-08
Gabriele Merziger: Approaches to Abductive Reasoning - An Overview -
46 pages

RR-92-09
Winfried Graf, Markus A. Thies:
Perspektiven zur Kombination von automatischem Animationsdesign und planbasierter Hilfe
15 Seiten

RR-92-11
Susane Biundo, Dietmar Dengler, Jana Koehler:
Deductive Planning and Plan Reuse in a Command Language Environment
13 pages

RR-92-13
Markus A. Thies, Frank Berger:
Planbasierte graphische Hilfe in objektorientierten Benutzungsoberflächen
13 Seiten

RR-92-14
Intelligent User Support in Graphical User Interfaces:
1. InCome: A System to Navigate through Interactions and Plans
   Thomas Fehrle, Markus A. Thies
2. Plan-Based Graphical Help in Object-Oriented User Interfaces
   Markus A. Thies, Frank Berger

22 pages

RR-92-15
Winfried Graf: Constraint-Based Graphical Layout of Multimodal Presentations
23 pages

RR-92-16
Jochen Heinsohn, Daniel Kudenko, Berhard Nebel, Hans-Jürgen Profitlich: An Empirical Analysis of Terminological Representation Systems
38 pages

RR-92-17
Hassan Aït-Kaci, Andreas Podelski, Gert Smolka:
A Feature-based Constraint System for Logic Programming with Entailment
23 pages

RR-92-18
John Nerbonne: Constraint-Based Semantics
21 pages

RR-92-19
Ralf Legleitner, Ansgar Bernardi, Christoph Klauck
PIM: Planning In Manufacturing using Skeletal Plans and Features
17 pages

RR-92-20
John Nerbonne: Representing Grammar, Meaning and Knowledge
18 pages

RR-92-22
Jörg Würtz: Unifying Cycles
24 pages

RR-92-24
Gabriele Schmidt: Knowledge Acquisition from Text in a Complex Domain
20 pages

DFKI Technical Memos

TM-91-09
Munindar P. Singh: On the Semantics of Protocols Among Distributed Intelligent Agents
18 pages

TM-91-10
Béla Buschauer, Peter Poller, Anne Schauder, Karin Harbusch: Tree Adjoining Grammars mit Unifikation
149 pages

TM-91-11
Peter Wazinski: Generating Spatial Descriptions for Cross-modal References
21 pages

TM-91-12
Klaus Becker, Christoph Klauck, Johannes Schwagereit: FEAT-PATR: Eine Erweiterung des D-PATR zur Feature-Erkennung in CAD/CAM
33 Seiten

TM-91-13
Knut Hinkelmann:
Forward Logic Evaluation: Developing a Compiler from a Partially Evaluated Meta Interpreter
16 pages

TM-91-14
Rainer Bleisinger, Rainer Hoch, Andreas Dengel:
ODA-based modeling for document analysis
14 pages

TM-91-15
Stefan Bussmann: Prototypical Concept Formation
An Alternative Approach to Knowledge Representation
28 pages

TM-92-01
Lijuan Zhang:
Entwurf und Implementierung eines Compilers zur Transformation von Werkstückrepräsentationen
34 Seiten

TM-92-02
Achim Schupeta: Organizing Communication and Introspection in a Multi-Agent Blocksworld
32 pages

DFKI Documents

D-91-10
Donald R. Steiner, Jürgen Müller (Eds.):
MAAMAW'91: Pre-Proceedings of the 3rd European Workshop on „Modeling Autonomous Agents and Multi-Agent Worlds"
246 pages
Note: This document is available only for a nominal charge of 25 DM (or 15 US-$).

D-91-11
*Thilo C. Horstmann:*Distributed Truth Maintenance
61 pages

D-91-12
Bernd Bachmann:
HieraCon - a Knowledge Representation System with Typed Hierarchies and Constraints
75 pages

D-91-13
International Workshop on Terminological Logics
Organizers: Bernhard Nebel, Christof Peltason, Kai von Luck
131 pages

D-91-14
Erich Achilles, Bernhard Hollunder, Armin Laux, Jörg-Peter Mohren: KRIS : Knowledge Representation and Inference System
- Benutzerhandbuch -
28 Seiten

D-91-15
Harold Boley, Philipp Hanschke, Martin Harm, Knut Hinkelmann, Thomas Labisch, Manfred Meyer, Jörg Müller, Thomas Oltzen, Michael Sintek, Werner Stein, Frank Steinle:
µCAD2NC: A Declarative Lathe-Worplanning Model Transforming CAD-like Geometries into Abstract NC Programs
100 pages

D-91-16
Jörg Thoben, Franz Schmalhofer, Thomas Reinartz:
Wiederholungs-, Varianten- und Neuplanung bei der Fertigung rotationssymmetrischer Drehteile
134 Seiten

D-91-17
Andreas Becker:
Analyse der Planungsverfahren der KI im Hinblick auf ihre Eignung für die Abeitsplanung
86 Seiten

D-91-18
Thomas Reinartz: Definition von Problemklassen im Maschinenbau als eine Begriffsbildungsaufgabe
107 Seiten

D-91-19
Peter Wazinski: Objektlokalisation in graphischen Darstellungen
110 Seiten

D-92-01
Stefan Bussmann: Simulation Environment for Multi-Agent Worlds - Benutzeranleitung
50 Seiten

D-92-02
Wolfgang Maaß: Constraint-basierte Plazierung in multimodalen Dokumenten am Beispiel des Layout-Managers in WIP
111 Seiten

D-92-03
Wolfgan Maaß, Thomas Schiffmann, Dudung Soetopo, Winfried Graf: LAYLAB: Ein System zur automatischen Plazierung von Text-Bild-Kombinationen in multimodalen Dokumenten
41 Seiten

D-92-06
Hans Werner Höper: Systematik zur Beschreibung von Werkstücken in der Terminologie der Featuresprache
392 Seiten

D-92-08
Jochen Heinsohn, Bernhard Hollunder (Eds.):
DFKI Workshop on Taxonomic Reasoning Proceedings
56 pages

D-92-09
Gernod P. Laufkötter: Implementierungsmöglichkeiten der integrativen Wissensakquisitionsmethode des ARC-TEC-Projektes
86 Seiten

D-92-10
Jakob Mauss: Ein heuristisch gesteuerter Chrat-Parser für attributierte Graph-Grammatiken
87 Seiten

D-92-15
DFKI Wissenschaftlich-Technischer Jahresbericht 1991
130 Seiten

D-92-21
Anne Schauder: Incremental Syntactic Generation of Natural Language with Tree Adjoining Grammars
57 pages

An Empirical Analysis of Terminological Representation Systems

Jochen Heinsohn, Daniel Kudenko, Bernhard Nebel, Hans-Jürgen Profitlich

RR-92-16
Research Report