High Performance Reasoning with Very Large Knowledge Bases: A Practical Case Study

Volker Haarslev and Ralf Möller
University of Hamburg, Computer Science Department
Vogt-Kölln-Str. 30, 22527 Hamburg, Germany
http://kogs-www.informatik.uni-hamburg.de/~haarslev|moeller/

Abstract

We present an empirical analysis of optimization techniques devised to speed up the so-called TBox classification supported by description logic systems which have to deal with very large knowledge bases (e.g. containing more than 100,000 concept introduction axioms). These techniques are integrated into the RACE architecture which implements a TBox and ABox reasoner for the description logic $\mathcal{ALCNH}_{R^+}$. The described techniques consist of adaptions of previously known as well as new optimization techniques for efficiently coping with these kinds of very large knowledge bases. The empirical results presented in this paper are based on experiences with an ontology for the Unified Medical Language System and demonstrate a considerable runtime improvement. They also indicate that appropriate description logic systems based on sound and complete algorithms can be particularly useful for very large knowledge bases.

1 Introduction

In application projects it is often necessary to deal with knowledge bases containing a very large number of axioms. Furthermore, many applications require only a special kind of axioms, so-called concept introduction axioms. Usually it has been argued that only systems based on incomplete calculi can deal with knowledge bases containing more than 100,000 axioms of this kind. In this contribution we present an empirical analysis of optimization techniques devised to improve the performance of description logic systems applied to this kind of knowledge bases. The analysis is based on the RACE[1] architecture [Haarslev and Möller, 2000a] which supports inference services for the description logic $\mathcal{ALCNH}_{R^+}$ [Haarslev and Möller, 2000b].[2]

As example knowledge bases we consider reconstructions of important parts of the UMLS (Unified Medical Language System) [McCray and Nelson, 1995] by using description logic representation techniques. The reconstruction is described in [Hahn *et al.*, 1999; Schulz and Hahn, 2000] and introduces a specific encoding scheme that uses several concept introduction axioms which represent subset as well as composition aspects of conceptual descriptions (words) mentioned in the UMLS metathesaurus.

Syntax	**Semantics**
Concepts	
A	$A^{\mathcal{I}} \subseteq \Delta_{\mathcal{I}}$ (A is a concept name)
$\neg C$	$\Delta_{\mathcal{I}} \setminus C^{\mathcal{I}}$
$C \sqcap D$	$C^{\mathcal{I}} \cap D^{\mathcal{I}}$
$C \sqcup D$	$C^{\mathcal{I}} \cup D^{\mathcal{I}}$
$\exists\, R\, .\, C$	$\{a \in \Delta_{\mathcal{I}} \mid \exists\, b \in \Delta_{\mathcal{I}} : (a,b) \in R^{\mathcal{I}}, b \in C^{\mathcal{I}}\}$
$\forall\, R\, .\, C$	$\{a \in \Delta_{\mathcal{I}} \mid \forall\, b \in \Delta_{\mathcal{I}} : (a,b) \in R^{\mathcal{I}} \Rightarrow b \in C^{\mathcal{I}}\}$
$\exists_{\geq n}\, S$	$\{a \in \Delta_{\mathcal{I}} \mid \lVert\{b \in \Delta_{\mathcal{I}} \mid (a,b) \in S^{\mathcal{I}}\}\rVert \geq n\}$
$\exists_{\leq m}\, S$	$\{a \in \Delta_{\mathcal{I}} \mid \lVert\{b \in \Delta_{\mathcal{I}} \mid (a,b) \in S^{\mathcal{I}}\}\rVert \leq m\}$
Roles	
R	$R^{\mathcal{I}} \subseteq \Delta_{\mathcal{I}} \times \Delta_{\mathcal{I}}$

$\| \cdot \|$ denotes the cardinality of a set, $S \in S$, $n, m \in \mathbb{N}$, $n > 0$.

TBox Axioms	
Syntax	Satisfied if
$R \in T$	$R^{\mathcal{I}} = (R^{\mathcal{I}})^+$
$R \sqsubseteq S$	$R^{\mathcal{I}} \subseteq S^{\mathcal{I}}$
$C \sqsubseteq D$	$C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$

ABox Assertions	
Syntax	Satisfied if
$a : C$	$a^{\mathcal{I}} \in C^{\mathcal{I}}$
$(a,b) : R$	$(a^{\mathcal{I}}, b^{\mathcal{I}}) \in R^{\mathcal{I}}$

Figure 1: Syntax and Semantics of $\mathcal{ALCNH}_{R^+}$.

The paper is structured as follows. We first introduce the syntax and semantics of $\mathcal{ALCNH}_{R^+}$ and characterize the form of axioms occurring in the UMLS knowledge bases. Afterwards we describe the following five complementary optimization techniques: (1) topological sorting for achieving a quasi definition order; (2) a method to cluster siblings in huge taxonomies; (3) a technique for efficiently addressing so-called domain and range restrictions; (4) exploitation of implicit disjointness declarations; (5) subset/superset caching for increasing runtime performance and reducing memory requirements. The effectiveness of these techniques is assessed using the empirical results obtained from processing the UMLS knowledge bases.

We briefly introduce the description logic (DL) $\mathcal{ALCNH}_{R^+}$ [Haarslev and Möller, 2000b] (see the tables in Figure 1) using a standard Tarski-style semantics. $\mathcal{ALCNH}_{R^+}$ extends the basic description logic $\mathcal{ALC}$ by

[1] URL: http://kogs-www.informatik.uni-hamburg.de/~race/

[2] A convenient pronunciation of $\mathcal{ALCNH}_{R^+}$ is ALC-nature.

role hierarchies, transitively closed roles (denoted by the set T in Figure 1), and number restrictions. Note that the combination of transitive roles and role hierarchies implies the expressivity of so-called general inclusion axioms (GCIs). The concept name $\top$ is used as an abbreviation for $\mathsf{C} \sqcup \neg\mathsf{C}$.

If R, S are role names, then $\mathsf{R} \sqsubseteq \mathsf{S}$ is called a *role inclusion* axiom. A *role hierarchy* $\mathcal{R}$ is defined by a finite set of role inclusion axioms. The concept language of $\mathcal{ALCNH}_{R^+}$ syntactically restricts the combinability of number restrictions and transitive roles due to a known undecidability result in case of an unrestricted syntax [Horrocks *et al.*, 1999]. Only simple roles may occur in number restrictions. Roles are called *simple* (denotes by the set S in Figure 1) if they are neither transitive nor have a transitive role as descendant.

If C and D are concept terms, then $\mathsf{C} \sqsubseteq \mathsf{D}$ *(generalized concept inclusion* or *GCI*) is a terminological axiom. A finite set $\mathcal{T}_{\mathcal{R}}$ of terminological axioms is called a *terminology* or *TBox* w.r.t. to a given role hierarchy $\mathcal{R}$.[3] A terminological axiom of the form $\mathsf{A} \sqsubseteq \mathsf{C}$ is called a *concept introduction axiom* and C is called the *primitive (concept) definition* of A if A is a concept name which occurs only once on the left hand side of the axioms contained in a TBox $\mathcal{T}$. A pair of GCIs of the form $\{\mathsf{A} \sqsubseteq \mathsf{C},\ \mathsf{C} \sqsubseteq \mathsf{A}\}$ (abbreviated as $\mathsf{A} \doteq \mathsf{C}$) is called a *concept definition axiom* and C is called the *(concept) definition* of A if A is a concept name which occurs only once on the left hand side of all axioms contained in a TBox $\mathcal{T}$.

An *ABox* $\mathcal{A}$ is a finite set of assertional axioms as defined in Figure 1. The ABox consistency problem is to decide whether a given ABox $\mathcal{A}$ is consistent w.r.t. a TBox $\mathcal{T}$ and a role hierarchy $\mathcal{R}$. An ABox $\mathcal{A}$ is consistent iff there exists a model $\mathcal{I}$ that satisfies all axioms in $\mathcal{T}$ and all assertions in $\mathcal{A}$. Subsumption between concepts can be reduced to concept satisfiability since C *subsumes* D iff the concept $\neg\mathsf{C} \sqcap \mathsf{D}$ is unsatisfiable. Satisfiability of concepts can be reduced to ABox consistency as follows: A concept C is satisfiable iff the ABox $\{\mathsf{a} : \mathsf{C}\}$ is consistent.

The DL reconstruction of important parts of the UMLS introduces a specific scheme where a set of concept introduction axioms is used to represent subset as well as composition aspects of conceptual descriptions (words) mentioned in the UMLS metathesaurus. For instance, for the notion of a 'heart', the following concept introduction axioms for heart structures (suffix 's'), heart parts (suffix 'p') and heart entities (no suffix) are declared (see [Schulz and Hahn, 2000] for details):

ana_heart $\sqsubseteq$ ana_heart_s $\sqcap$ ana_hollow_viscus $\sqcap$
umls_body_part_organ_or_organ_component

ana_heart_s $\sqsubseteq$ ana_hollow_viscus_s $\sqcap$
ana_cardiovascular_system_p

ana_heart_p $\sqsubseteq$ $\neg$ana_heart $\sqcap$ ana_heart_s $\sqcap$
$\exists_{\geq 1}$ anatomical_part_of_ana_heart

Note the implicit disjointness declared between ana_heart_p and ana_heart. The following role axiom is generated as well.

anatomical_part_of_ana_heart $\sqsubseteq$
anatomical_part_of_ana_hollow_viscus

It is beyond the scope of this paper to discuss the pros and cons of specific modeling techniques used in the UMLS reconstruction. In the next section, optimization techniques for efficiently dealing with these kinds of knowledge bases are presented.

2 Optimization Techniques

Modern DL systems such as RACE offer at least two standard inference services for concept names occurring in TBoxes: classification and coherence checking. Classification is the process of computing the most-specific subsumption relationships between all concept names mentioned in a TBox $\mathcal{T}$. The result is often referred to as the *taxonomy* of $\mathcal{T}$ and gives for each concept name two sets of concept names listing its "parents" (direct subsumers) and "children" (direct subsumees). Coherence checking determines all concept names which are unsatisfiable.

Expressive DLs such as $\mathcal{ALCNH}_{R^+}$ allow GCIs which can considerably slow down consistency tests. A *true* GCI is a GCI of the form $\mathsf{C} \sqsubseteq \mathsf{D}$ where C is not a name and $\mathsf{C} \sqsubseteq \mathsf{D}$ is not part of a pair representing a concept definition axiom. A standard technique (GCI absorption) [Horrocks and Tobies, 2000] which is also part of the RACE architecture performs a TBox transformation which precedes classification and coherence checking. The axioms in a TBox are transformed in a way that true GCIs can be absorbed into (primitive) concept definitions which can be efficiently dealt with by a technique called *lazy unfolding* (e.g. see [Baader *et al.*, 1994]). Lazy unfolding dynamically expands a concept name by its (primitive) definition during an ABox consistency test. The true GCIs remaining after the GCI absorption are referred to as *global axioms*.

Our findings indicate that state-of-the-art techniques currently employed for fast classification of TBoxes have to be extended in order to cope with very large knowledge bases of the above-mentioned kind. In the following we describe these extensions.

2.1 Topological Sorting for Quasi Definition Order

For TBox classification the RACE architecture employs the techniques introduced in [Baader *et al.*, 1994]. The parents and children of a certain concept name are computed in so-called 'top search' and 'bottom search' traversal phases, respectively. These phases can be illustrated with the following example. Assume a new concept name $\mathsf{A_i}$ has to be inserted into an existing taxonomy. The top search phase traverses the taxonomy from the top node ($\top$) via the set of children and checks whether $\mathsf{A_i}$ is subsumed by a concept node. Basically, if $\mathsf{A_i}$ is subsumed by a node $\mathsf{A_j}$, then the children of $\mathsf{A_j}$ are traversed. The top search phase determines the set of parents of $\mathsf{A_i}$. When the top search phase for $\mathsf{A_i}$ has been finished, the bottom search phase for $\mathsf{A_i}$ analogously traverses the taxonomy from the bottom node via the set of parents of a node. The bottom search phase determines the set of children of $\mathsf{A_i}$.

Using the marking and propagation techniques described in [Baader *et al.*, 1994] the search space for traversals can usually be pruned considerably. It is always advantageous to

[3] The reference to $\mathcal{R}$ is omitted in the following if we use $\mathcal{T}$.

avoid as many traversals as possible since they require the use of "expensive" subsumption tests. This is even more important for very large TBoxes.

Let us assume, a TBox to be classified can be transformed such that no global axioms remain but cyclic (primitive) concept definitions may exist. According to [Baader *et al.*, 1994] we assume that a concept name A 'directly uses' a concept name B if B occurs in the (primitive) definition of A. The relation 'uses' is the transitive closure of 'directly uses.' If A uses B then B comes before A in the so-called definition order. For acyclic TBoxes (i.e. the uses relation is irreflexive) containing concept introduction axioms only, the set of concept names can be processed in definition order, i.e. a concept name is not classified until all the concept names used in its (primitive) definition are classified. In this case the set of children of a concept name to be inserted consists only of the bottom concept. Thus, if concept names are classified in definition order, the bottom search phase can safely be omitted for concept names which have only a primitive definition [Baader *et al.*, 1994].

In order to avoid the bottom search phase it is possible to impose a syntactical restriction on TBoxes for less expressive DLs, i.e. to accept only concept axioms in definition order, i.e. the (primitive) definitions do not include forward references to concept names not yet introduced. However, for an expressive DL such as $\mathcal{ALCNH}_{R^+}$, which offers cyclic axioms and GCIs, in general, the bottom search phase cannot be skipped.

The UMLS TBoxes contain many forward references occurring in value (e.g. $\forall\,\mathsf{R}\,.\,\mathsf{C}$) and existential restrictions (e.g. $\exists\,\mathsf{R}\,.\,\mathsf{C}$). Thus, the definition order of concept names has to be computed in a preprocessing step. As a refinement we devised a slightly less strict notion of definition order which works more efficiently. We assume a relation 'directly refers to' similar to 'directly uses' but with references occurring in the scope of quantifiers not considered. This simplification reduces the overhead caused by computing the 'directly refers to' relation. It is correct since subsumption between concept names with primitive definitions cannot be caused via quantifiers occurring in the concept definitions. Again 'refers to' is the transitive closure of 'directly refers to'. The 'refers to' relation induces a partial order relation on sets of concept names. All concept names involved in a cycle become members of one set while the remaining concept names form singleton sets. A topological sorting algorithm is used to serialize the partial order such that a total order between sets of concept names is defined. This serialization is called a *quasi definition order*.

During the classification of a TBox with RACE the sets of concept names are processed in quasi definition order. For each singleton set whose member has a primitive concept definition, the bottom search can be skipped. Let $\mathsf{A} \sqsubseteq \mathsf{C}$ be a concept introduction axiom and A is to be inserted into the taxonomy. The 'refers to' relation and the quasi definition order serialization ensure that either all concept names that are potential subsumees of A are inserted after A has been inserted into the subsumption lattice or the bottom search is indeed performed. The quasi definition order is conservative w.r.t. the potential subsumers (note that $\mathcal{ALCNH}_{R^+}$ does not support inverse roles). Moreover, in a basic subsumption test the subsumption lattice under construction is never referred to. Thus, strict definition order classification is not necessary.

Note that in [Baader *et al.*, 1994] no experiments are discussed that involve the computation of a serialization given a TBox with axioms not already in (strict) definition order. Topological sorting is of order $n+e$ where e is the number of given 'refers to' relationships. Thus, we have approximately O($n \log n$) steps while the bottom search procedure requires O(n^2) steps in the worst case.

2.2 Adaptive Clustering in TBoxes

Experiments with the UMLS TBoxes showed that the well-known techniques described in [Baader *et al.*, 1994] exhibit considerable performance deficiencies in case of (rather flat) taxonomies where some nodes have a large number of children. Therefore, in the RACE architecture a special clustering technique is employed.

If the top search phase finds a node (e.g. A) with more than θ children, the θ children are grouped into a *bucket* (e.g. A_{new}), i.e. a (virtual) concept definition axiom $\mathsf{A}_{new} \doteq \mathsf{A}_1 \sqcup \ldots \sqcup \mathsf{A}_\theta$ is assumed and A_{new} is inserted into the subsumption lattice with $\{\mathsf{A}\}$ being its parents and $\{\mathsf{A}_1 \ldots \mathsf{A}_\theta\}$ being its children. A_{new} is also referred to as a bucket concept. Note that bucket concepts (e.g. A_{new}) are considered as virtual in the sense that the external interface of RACE hides the names of virtual concepts in a transparent way.

Let us assume, a certain concept name B is to be inserted and a node A with its children $\{\mathsf{A}_1, \ldots, \mathsf{A}_\theta\}$ is encountered during the top search phase. Instead of testing for each child A_i ($\mathsf{i} \in \{1..\theta\}$) whether A_i subsumes B, our findings suggest that it is more effective to initially test whether the associated virtual concept definition of A_{new} does not subsume B using the pseudo model merging technique (e.g. see [Horrocks, 1997; Haarslev *et al.*, 2001]) which provides a 'cheap' and sound but incomplete non-subsumption test based on pseudo models derived from concept satisfiability tests. Since in most cases no subsumption relation can be found between any A_i and B, one test possibly replaces θ "expensive" but wasted subsumption tests. On the other hand, if a subsumption relation indeed exists, then clustering introduces some overhead. However, since the UMLS TBoxes mostly contain concept introduction axioms, the pseudo model of $\neg\mathsf{A}_{new}$ being used for model merging is very simple because the pseudo model basically consists only of a set of negated primitive concept names (see [Haarslev *et al.*, 2001] for further details about pseudo model merging in RACE). The adaptive clustering is designed in a way that it still works even if a quasi definition order cannot be guaranteed (e.g. due to the presence of defined concepts or GCIs). Therefore, a bucket becomes obsolete and has to be removed from a concept node if a member of this bucket gets a different concept node as parent.

For best performance, the number of children to be kept in a bucket should depend on the total number of known children for a concept. However, this can hardly be estimated. Therefore, the following adaptive strategy is used. If more and more concept names are "inserted" into the subsumption lattice, the number of buckets increases as well. If a new bucket is to be created for a certain node A and there are already σ buckets clustering the children of A, then two buckets (those buckets with the smallest number of children) of A are merged. For instance, merging of the buckets $\mathsf{A}_{\mathsf{new}} \doteq \mathsf{A}_1 \sqcup \ldots \sqcup \mathsf{A}_\mathsf{n}$ and $\mathsf{B}_{\mathsf{new}} \doteq \mathsf{B}_1 \sqcup \ldots \sqcup \mathsf{B}_\mathsf{m}$ means that the bucket $\mathsf{A}_{\mathsf{new}}$ is "redefined" as $\mathsf{A}_{\mathsf{new}} \doteq \mathsf{A}_1 \sqcup \ldots \sqcup \mathsf{A}_\mathsf{n} \sqcup \mathsf{B}_1 \sqcup \ldots \sqcup \mathsf{B}_\mathsf{m}$ and the bucket $\mathsf{B}_{\mathsf{new}}$ is reused for the new bucket to be created (see above).[4] Whether hierarchical clustering techniques lead to performance improvements is subject to further research.

The current evaluation of clustering with buckets uses a setting with $\theta = 10$ and $\sigma = 15$.

2.3 Dealing with Domain and Range Restrictions

Some UMLS TBoxes contain axioms declaring domain and range restrictions for roles. For instance, the *domain* C of a role R can be determined by the axiom $\exists_{\geq 1} \mathsf{R} \sqsubseteq \mathsf{C}$ and the *range* D by the axiom $\top \sqsubseteq \forall \mathsf{R} . \mathsf{D}$. Domain restrictions increase the search space during a consistency test since they have to be represented as disjunctions of the form $(\neg \exists_{\geq 1} \mathsf{R}) \sqcup \mathsf{C}$.

It is possible to transform a domain restriction of the form $\exists_{\geq 1} \mathsf{R} \sqsubseteq \mathsf{C}$ into an equivalent inclusion axiom of the form $\neg \mathsf{C} \sqsubseteq \exists_{\leq 0} \mathsf{R}$ and to absorb the transformed axiom if no inclusion axiom of the form $\mathsf{C} \sqsubseteq \ldots$ exists. However, the transformation is not applicable to the UMLS TBoxes since they contain domain restrictions (e.g. $\exists_{\geq 1}$ $\mathsf{anatomical\_part\_of\_ana\_heart} \sqsubseteq \mathsf{ana\_heart\_p}$) as well as inclusion axioms (e.g. $\mathsf{ana\_heart\_p} \sqsubseteq \neg \mathsf{ana\_heart} \sqcap \mathsf{ana\_heart\_s} \sqcap \exists_{\geq 1}$ $\mathsf{anatomical\_part\_of\_ana\_heart}$) violating the above-mentioned precondition. Hence, in order to apply the topological sorting optimization, it was necessary to incorporate domain restrictions into an ABox consistency test because global axioms may not exist in order to apply the topological sorting technique.

If GCIs representing domain restrictions for roles have been absorbed, they are dealt with by RACE with a generalized kind of lazy unfolding. Whenever a concept of the form $\exists \mathsf{R} . \mathsf{D}$ or $\exists_{\geq n} \mathsf{R}$ is checked for unfolding, it is replaced by $\mathsf{C} \sqcap \exists \mathsf{R} . \mathsf{D}$ or $\mathsf{C} \sqcap \exists_{\geq n} \mathsf{R}$ if a GCI of the form $\exists_{\geq 1} \mathsf{S} \sqsubseteq \mathsf{C}$ has been absorbed (R a sub-role of S). This technique is valid since $\exists_{\geq 1} \mathsf{R} \sqsubseteq \mathsf{C}$ can be represented as the global axiom $\exists_{\leq 0} \mathsf{R} \sqcup \mathsf{C}$ and the unfolding scheme of RACE guarantees that C is added if an R-successor will be created due to $\exists \mathsf{R} . \mathsf{D}$ or $\exists_{\geq n} \mathsf{R}$. A role assertion $(\mathsf{a}, \mathsf{b}) : \mathsf{R}$ is unfolded to $\{(\mathsf{a}, \mathsf{b}) : \mathsf{R}, \mathsf{a} : \mathsf{C}\}$. If lazy unfolding is applied, domain restrictions have to be considered w.r.t. the 'directly refers to' relationship in a special way.

Note that, in principle, RACE also supports the absorption of GCIs of the form $\neg \mathsf{A} \sqsubseteq \mathsf{C}_1$ (but only if no concept introduction axiom of the form $\mathsf{A} \sqsubseteq \mathsf{C}_2$ and no concept definition definition axiom of the form $\mathsf{A} \doteq \mathsf{C}_2$ for A exists). Some knowledge bases can only be handled effectively if the absorption of axioms of the form $\neg \mathsf{A} \sqsubseteq \mathsf{C}_1$ is supported.

In contrast to domain restrictions, range restrictions for roles do not introduce new disjunctions. However, it is always advantageous to keep the number of global axioms to be managed as small as possible. Therefore, GCIs expressing range restrictions for roles are absorbed and concepts of the form $\exists \mathsf{R} . \mathsf{C}$ or $\exists_{\geq n} \mathsf{R}$ are unfolded to $\exists \mathsf{R} . \mathsf{C} \sqcap \forall \mathsf{S} . \mathsf{D}$ or $\exists_{\geq n} \mathsf{R} \sqcap \forall \mathsf{S} . \mathsf{D}$ if a GCI of the form $\top \sqsubseteq \forall \mathsf{S} . \mathsf{D}$ (R a sub-role of S) has been absorbed. A role assertion $(\mathsf{a}, \mathsf{b}) : \mathsf{R}$ is unfolded to $\{(\mathsf{a}, \mathsf{b}) : \mathsf{R}, \mathsf{b} : \mathsf{D}\}$.

2.4 Exploiting Disjointness Declarations

As has been discussed in [Baader *et al.*, 1994], it is important to derive told subsumers[5] for each concept name for marking and propagation processes. Besides told subsumers, RACE exploits also the set of "told disjoints[6]". In the 'heart' example presented above, $\mathsf{ana\_heart}$ is computed as a told disjoint concept of $\mathsf{ana\_heart\_p}$ by examining the related concept introduction axioms. If it is known that a concept B is a subsumer of a concept A then A cannot be a subsumee of the told disjoints of B. This kind of information is recorded (and propagated) with appropriate non-subsumer marks (see [Baader *et al.*, 1994] for details about marking and propagation operations) in order to prune the search space for traversals causing subsumption tests. Exploiting disjointness information has not been investigated in [Baader *et al.*, 1994].

2.5 Caching Policies

RACE supports different caching policies (see also [Haarslev and Möller, 2000a] for caching in RACE). Two types of caches are provided which can be used together or alternatively. Both cache types are accessed via keys constructed from a set of concepts. The first cache (called equal cache) contains entries about the satisfiability status of concept conjunctions already encountered. This cache only returns a hit if the search key exactly matches (i.e. is equal to) the key of a known entry. For instance, the key for a concept conjunction $\mathsf{C}_1 \sqcap \ldots \sqcap \mathsf{C}_\mathsf{n}$ is the (ordered) set $\{\mathsf{C}_1, \ldots, \mathsf{C}_\mathsf{n}\}$ of concepts.

The second cache type consists of a pair of caches. The subset cache contains only entries for satisfiable concept conjunctions while the superset cache stores unsatisfiable concept conjunctions. These caches support queries concerning already encountered supersets and subsets of a given search key (see also [Hoffmann and Köhler, 1999; Giunchiglia and Tacchella, 2000]). For instance, if the subset (satisfiability) cache already contains an entry for the key $\{\mathsf{C}_1, \mathsf{C}_2, \mathsf{C}_3\}$ and is queried with the key $\{\mathsf{C}_1, \mathsf{C}_3\}$, it returns a hit, i.e. the conjunction $\mathsf{C}_1 \sqcap \mathsf{C}_3$ is also satisfiable. Analogously, if the superset (unsatisfiability) cache already contains an entry for the key $\{\mathsf{D}_2, \mathsf{D}_3, \mathsf{D}_5\}$ and is queried with the key $\{\mathsf{D}_1, \ldots, \mathsf{D}_6\}$, it returns a hit, i.e. the conjunction $\mathsf{D}_1 \sqcap \ldots \sqcap \mathsf{D}_6$ is also unsatisfiable. If the equal cache is enabled, it is the first reference, i.e. only if an equal cache lookup fails, the superset or subset caches are consulted.

[4] Note that due to subsequent merging operations, n and m need not be equal to θ.

[5] For instance, $\mathsf{A}_1, \mathsf{A}_2$ are told subsumers of A for a concept introduction axiom $\mathsf{A} \sqsubseteq \mathsf{A}_1 \sqcap \mathsf{A}_2$ if $\mathsf{A}_1, \mathsf{A}_2$ are concept names.

[6] For instance, $\mathsf{A}_1, \mathsf{A}_2$ are told disjoints of A for a concept introduction axiom $\mathsf{A} \sqsubseteq \neg \mathsf{A}_1 \sqcap \neg \mathsf{A}_2$ if $\mathsf{A}_1, \mathsf{A}_2$ are concept names.

3 Empirical Results for UMLS Classifications

The performance of the RACE system is evaluated with different versions of the UMLS knowledge bases. UMLS-1 is a preliminary version whose classification resulted in many unsatisfiable concept names. UMLS-1 contains approximately 100,000 concept names and for almost all of them there exists a concept introduction axiom of the form $\mathsf{A} \sqsubseteq \mathsf{C}$ where C not equal to $\top$. In addition, in UMLS-1 80,000 role names are declared. Role names are arranged in a hierarchy.

UMLS-2 is a new version in which the reasons for the inconsistencies have been removed. The version of UMLS-2 we used for our empirical tests uses approximately 160,000 concept names with associated primitive concept definitions and 80,000 hierarchical roles.

Originally, the UMLS knowledge bases have been developed with an incomplete description logic system which does not classify concepts with cyclic definitions (and, in turn, the concepts which use these concepts). Due to the treatment of cycles in the original approach [Schulz and Hahn, 2000], the cycle-causing concepts are placed in so-called `:implies` clauses, i.e. these concepts are not considered in concept satisfiability and subsumption tests. For the same reason, the UMLS reconstruction uses `:implies` for domain and range restrictions of roles, i.e. domain and range restrictions are also not considered in concept satisfiability and subsumption tests.

With RACE, none of these pragmatic distinctions are necessary. However, in order to mimic the original behavior and to test more than one TBox with RACE, for each of the knowledge base versions, UMLS-1 and UMLS-2, three different subversions are generated (indicated with letters a, b and c). Version 'a' uses axioms of the style presented above, i.e. the `:implies` parts are omitted for TBox classification (and coherence checking). In version 'b' the `:implies` part of the original knowledge base is indeed considered for classification by RACE. Thus, additional axioms of the following form are part of the TBox.

$$\begin{aligned}\mathbf{ana\_heart} \sqsubseteq\ & \exists\, \mathsf{has\_developmental\_fo} \,.\, \mathsf{ana\_fetal\_heart} \sqcap \\ & \exists\, \mathsf{surrounded\_by} \,.\, \mathsf{ana\_pericardium}\end{aligned}$$

Version 'c' is the hardest version. Additional axioms express domain and range restrictions for roles. For instance, the following axioms are included in the TBox for anatomical_part_of_ana_heart.

$$\exists_{\geq 1}\ \mathsf{anatomical\_part\_of\_ana\_heart} \sqsubseteq \mathsf{ana\_heart\_p}$$

$$\top \sqsubseteq \forall\, \mathsf{anatomical\_part\_of\_ana\_heart} \,.\, \mathsf{ana\_heart}$$

Thus, for the performance evaluation we have tested 6 different knowledge bases. All measurements have been performed on a Sun UltraSPARC 2 with 1.4 GByte main memory and Solaris 6. RACE is implemented in ANSI Common Lisp and for the tests Franz Allegro Common Lisp 5.0.1 has been used. The results are as follows.

If the generalized unfolding technique for domain and range restrictions is disabled in RACE, even a small subset (with a size of ∼5%) of the UMLS TBoxes with GCIs for domain and range restrictions could not be classified within several hours of runtime. Furthermore, the equal cache had to be disabled for all UMLS TBoxes in order to reduce the space requirements during TBox classification.

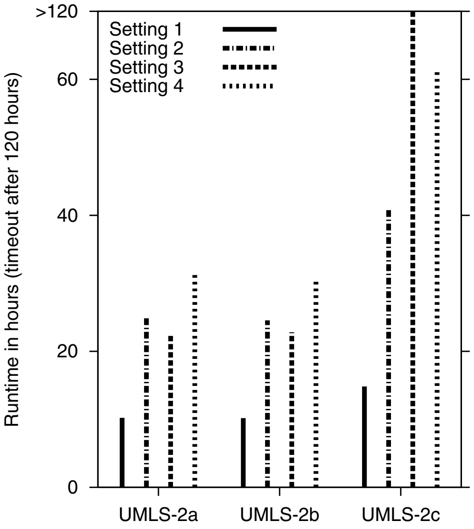

Figure 2: Evaluation of the topological sorting and clustering techniques for UMLS2a-c (see explanation in text).

Without clustering and topological sorting, UMLS-1a can be classified in approximately 11 hours (1636 concept names are recognized as unsatisfiable). With clustering and topological sorting enabled, only 5.5 hours are necessary to compute the same result for UMLS-1a. The second version, UMLS-1b, requires 3.6 hours (with optimization) and 6.1 hours (without optimization). The reason for the enhanced performance with more constraints is that in this version already 47855 concept names are unsatisfiable. With domain and range restrictions we found that even 60246 concept names are unsatisfiable. The computation times with RACE are 3.4 hours (with optimization) and 8.7 hours (without optimization). Up to 500 MBytes of memory are required to compute the classification results. For UMLS-1, checking TBox coherence (see above) requires approximately 10 minutes.

The new second version, UMLS-2, contains an additional part of the UMLS ontology and, therefore, is harder to deal with. Furthermore, there are no unsatisfiable concept names, i.e. classification is much harder because there are much more nodes in the subsumption lattice. In UMLS-1, due to the large number of unsatisfiable concepts, the subsumption lattice is rather small because many concept names "disappear" as synonyms of the bottom concept. For UMLS-2, checking TBox coherence (see above) requires between 15 and 50 minutes (2a: 16 min, 2b: 19 min, 2c: 51 min).

The results for classifying the UMLS-2 TBoxes are shown in Figure 2. A comparison of setting 1 (all optimizations enabled) and setting 2 (clustering disabled) reveals that clustering is a very effective optimization technique for the UMLS-2 TBoxes. The result for setting 3 (topological sorting disabled) and UMLS-2a/b supports the fact that topological sorting is also very effective. The runtime result for setting 3 and UMLS-2c is due to removed buckets (see Section 2.2). This is very likely to happen if topological sorting is disabled and apparently shows a dramatic overhead for UMLS-2c. If, in setting 4, both clustering and topological sorting are disabled, runtimes increase only to a limited extent. Moreover, according to the evaluation results, UMLS-2b does not require more computational resources than UMLS-2a (see the discussion about `:implies` from above). Only the incorporation of domain and range restrictions cause runtimes to increase. For the UMLS-2 TBoxes up to 800 MBytes of memory are required. For other benchmark TBoxes (e.g. Galen [Horrocks, 1997] with approx. 3000 concepts) our results indicate an overhead of ∼5% in runtime. This is caused by the presence of many defined concepts and a small average number of concept children in the taxonomy. In summary, the results for the UMLS TBoxes clearly demonstrate that clustering is only effective in conjunction with topological sorting establishing a quasi-definition order.

4 Conclusion

In this paper enhanced optimization techniques which are essential for an initiative towards sound and complete high performance knowledge base classification are presented. Thus, fast classification of very large terminologies which mostly consist of concept introduction axioms now has become possible with description logic systems based on sound and complete algorithms.

A final comment concerning the significance of the UMLS knowledge bases used for the empirical evaluation is appropriate. Even though the UMLS knowledge bases do not make use of defined concepts or arbitrary GCIs, a large number of concept introduction axioms and a possibly large role hierarchy can be called the standard case in many practical applications. Furthermore, some UMLS TBoxes (version 'c') demand the absorption of GCIs expressing domain and range restrictions for roles. Without this new technique even a very small subset of these TBoxes cannot be classified within a reasonable amount of time. If description logics are to be successful in large-scale practical applications, being able to deal with very large knowledge bases such as those based on the UMLS is mandatory.

Even with the above-mentioned optimization techniques, dealing with 160,000 concept names is by no means a trivial task. Large knowledge bases reveal even slightly less than optimal algorithms used for specific subproblems. Thus, experiments with very large knowledge bases also provide feedback concerning lurking performance bottlenecks not becoming apparent when dealing with smaller knowledge bases. To the best of our knowledge, RACE is the only DL system based on sound and complete algorithms that can deal with this kind of very large knowledge bases.

We would like to thank Stefan Schulz for making the UMLS reconstruction available.

References

[Baader *et al.*, 1994] F. Baader, E. Franconi, B. Hollunder, B. Nebel, and H.J. Profitlich. An empirical analysis of optimization techniques for terminological representation systems or: Making $\mathcal{KRIS}$ get a move on. *Applied Artificial Intelligence. Special Issue on Knowledge Base Management*, 4:109–132, 1994.

[Cohn *et al.*, 2000] A.G. Cohn, F. Giunchiglia, and B. Selman, editors. *Proceedings of Seventh International Conference on Principles of Knowledge Representation and Reasoning (KR'2000), Breckenridge, Colorado, USA, April 11-15, 2000*, April 2000.

[Giunchiglia and Tacchella, 2000] E. Giunchiglia and A. Tacchella. A subset-matching size-bounded cache for satisfiability of modal logics. In *Proceedings International Conference Tableaux'2000*, pages 237–251. Springer-Verlag, 2000.

[Haarslev and Möller, 2000a] V. Haarslev and R. Möller. Consistency testing: The RACE experience. In R. Dyckhoff, editor, *Proceedings, Automated Reasoning with Analytic Tableaux and Related Methods, University of St Andrews, Scotland, 4-7 July, 2000*, pages 57–61. Springer-Verlag, Berlin, April 2000.

[Haarslev and Möller, 2000b] V. Haarslev and R. Möller. Expressive ABox reasoning with number restrictions, role hierarchies, and transitively closed roles. In Cohn et al. [2000], pages 273–284.

[Haarslev *et al.*, 2001] V. Haarslev, R. Möller, and A.-Y. Turhan. Exploiting pseudo models for TBox and ABox reasoning in expressive description logics. In *Proceedings of the International Joint Conference on Automated Reasoning, IJCAR'2001, June 18-23, 2001, Siena, Italy*, LNCS. Springer-Verlag, Berlin, June 2001.

[Hahn *et al.*, 1999] U. Hahn, S. Schulz, and M. Romacker. Part-whole reasoning: A case study in medical ontology engineering. *IEEE Intelligent Systems*, 14(5):59–67, September 1999.

[Hoffmann and Köhler, 1999] J. Hoffmann and J. Köhler. A new method to query and index sets. In *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence IJCAI-99*, pages 462–467. Morgan-Kaufmann Publishers, July 31 – August 6, 1999.

[Horrocks and Tobies, 2000] I. Horrocks and S. Tobies. Reasoning with axioms: Theory and practice. In Cohn et al. [2000], pages 285–296.

[Horrocks *et al.*, 1999] I. Horrocks, U. Sattler, and S. Tobies. Practical reasoning for expressive description logics. In *Proceedings of the 6th International Conference on Logic for Programming and Automated Reasoning (LPAR'99)*, pages 161–180. Springer-Verlag, Berlin, September 1999.

[Horrocks, 1997] I. Horrocks. *Optimising Tableaux Decision Procedures for Description Logics*. PhD thesis, University of Manchester, 1997.

[McCray and Nelson, 1995] A.T. McCray and S.J. Nelson. The representation of meaning in the UMLS. *Methods of Information in Medicine*, 34(1/2):193–201, 1995.

[Schulz and Hahn, 2000] S. Schulz and U. Hahn. Knowledge engineering by large-scale knowledge reuse – Experience from the medical domain. In Cohn et al. [2000], pages 601–610.