Speeding up Approximation with nicer Concepts

Anni-Yasmin Turhan* and Yusri Bong

Theoretical Computer Science, TU Dresden, Germany
{turhan, bong}@tcs.inf.tu-dresden.de

Abstract. Concept approximation is an inference service for Description Logics that provides "translations" of concept descriptions from one DL to a less expressive DL. In [4] a method for optimizing the computation of $\mathcal{ALC}$-$\mathcal{ALE}$-approximations of $\mathcal{ALC}$-concept descriptions was introduced. The idea is to characterize a certain class of concept descriptions for which conjuncts can be approximated independently. In this paper we provide relaxed conditions for this class of $\mathcal{ALC}$-concept descriptions, extend this notion to number restrictions and report on a first implementation of this method for $\mathcal{ALCN}$-$\mathcal{ALEN}$-approximation.

1 Motivation

Approximation is a non-standard inference service in Description Logics (DLs) introduced in [3]. Approximating a concept description, defined in one DL, means to translate this concept description to another concept description, defined in a second, typically less expressive DL, such that both concepts are as closely related as possible with respect to subsumption. Like other non-standard inferences such as computing the least common subsumer (lcs) or matching of concepts, approximation has been introduced to support the construction and maintenance of DL knowledge bases.

The building of ontologies can be assisted by employing the bottom-up approach, where new concepts can be derived from a collection of already existing ones by computing their commonalities. This task is typically realized by computing the lcs. If the employed DL provides disjunction, the lcs is just the disjunction of the input concepts. Thus, a user inspecting this concept does not learn anything about their commonalities. By using concept approximation, however, one can make the commonalities explicit to some extent by first approximating the concepts in a sub-language which does not provide disjunction, and then computing the lcs of the approximations. Besides this approximation is used to compute semantically closely related versions of expressive knowledge bases to port knowledge bases between different systems or to integrate different knowledge bases. Moreover, users who are no DL experts can be supported in comprehending a knowledge base written in an expressive DL by offering them a simplified view of it obtained by approximation.

A worst case double exponential algorithm for approximating $\mathcal{ALCN}$- by $\mathcal{ALEN}$-concept descriptions was presented in [2]. A first implementation of approximation presented in [3] revealed that run-times for concepts of moderate size

* This work was partially supported by EU FET TONES (grant IST-2005-7603).

is already a couple of seconds. For an interactive application as the bottom-up construction faster run-times are desirable. In [4] the following approach to optimize computation of approximations was presented: Instead of approximating a concept C as a whole, a significant amount of time could be saved by splitting C into its conjuncts and approximating them separately. If, for instance, C consists of two conjuncts of size n then the approximation of C takes some $a^{b^{2n}}$ steps while the conjunct-wise approach would just take $2a^{b^n}$. Unfortunately, splitting an arbitrary input concept at conjunctions leads to incorrect approximations. A class for which this conjunct-wise approximation still produces the correct result is the class of so-called *nice concepts*. Moreover, a characterization of nice concepts is also an important prerequisite for another way of optimizing computations of concept approximation, namely, non-naive use of caching. Say, we want to approximate $C \sqcap D$, where C and D are complex $\mathcal{ALCN}$-concept descriptions. Now, if $C \sqcap D$ is nice and we have cached the approximation of C, we only need to compute the approximation of D and conjoin it with the cached result, instead of computing the whole approximation from scratch.

2 Preliminaries

We assume that the reader is familiar with the basic notions of DLs, see [1]. *Concept descriptions* are inductively defined based on a set of *concept constructors* starting with a set N_C of *concept names* and a set N_R of *role names*. The DL $\mathcal{ALE}$ provides the constructors conjunction, existential and value restrictions as well as primitive negation, i.e., only concept names can be negated. $\mathcal{ALC}$ extends $\mathcal{ALE}$ by full negation and disjunction. $\mathcal{ALCN}$ ($\mathcal{ALEN}$) adds number restrictions to $\mathcal{ALC}$ ($\mathcal{ALE}$). For the syntax and model theoretic semantics of the mentioned concept constructors, see [1]. In addition to the usual definition, we require that TBoxes are *unfoldable*, i.e., their concept definitions are acyclic and unique. In order to approximate $\mathcal{ALCN}$-concept descriptions by $\mathcal{ALEN}$-concept descriptions, we need to compute the lcs in $\mathcal{ALEN}$.

Definition 1 (lcs). *Given $\mathcal{ALEN}$-concept descriptions $C_1, \ldots, C_n$ with $n \geq 2$, the $\mathcal{ALEN}$-concept description C is the* least common subsumer *(lcs) of $C_1, \ldots, C_n$ iff (i) $C_i \sqsubseteq C$ for all $1 \leq i \leq n$, and (ii) C is the least concept description with this property, i.e., if C' satisfies $C_i \sqsubseteq C'$ for all $1 \leq i \leq n$, then $C \sqsubseteq C'$.*

As already mentioned, in $\mathcal{ALCN}$ the lcs trivially exists since $lcs(C, D) \equiv C \sqcup D$. To obtain a more meaningful concept description, we first approximate the $\mathcal{ALCN}$-concept descriptions and then compute their lcs.

Definition 2 (approximation). *Let $\mathcal{L}_1$ and $\mathcal{L}_2$ be two DLs, and let C be an $\mathcal{L}_1$- and D be an $\mathcal{L}_2$-concept description. Then, D is called an $\mathcal{L}_1 - \mathcal{L}_2$-*approximation *of C (written $D = approx_{\mathcal{L}_2}(C)$) iff (i) $C \sqsubseteq D$, and (ii) D is minimal with this property, i.e., $C \sqsubseteq D'$ and $D' \sqsubseteq D$ implies $D' \equiv D$ for all $\mathcal{L}_2$-concept descriptions D'.*

Intuitively, an approximation of an $\mathcal{ALCN}$-concept description is an $\mathcal{ALEN}$-concept description that is more general than the input concept description but minimal w.r.t. subsumption.

2.1 Computation Algorithm for $\mathcal{ALCN}$-$\mathcal{ALEN}$-approximations

We sketch the computation algorithm for $\mathcal{ALCN}$-$\mathcal{ALEN}$-approximations briefly—for its exact definition refer to [2]. In case an approximation of an $\mathcal{ALCN}$-concept C that uses names defined in a TBox is to be computed, then these names have to be first replaced by their definition. If C is equivalent to $\top$ ($\bot$), its approximation is $\top$ ($\bot$), otherwise the concept description is normalized. First, the concept description is transformed into negation normal form (NNF). Second, the obtained concept description is transformed into $\mathcal{ALCN}$-normal form ($\mathcal{ALCN}$-NF). In this step conjunctions are distributed over the disjunctions. In order to describe the disjuncts obtained by the $\mathcal{ALCN}$-NF, some notation is needed to access the different parts of a concept description C.

- $prim(C)$ denotes the set of all (negated) concept names and $\bot$ occurring on the top-level of C;
- $\mathsf{val}_r(C) := C_1 \sqcap \cdots \sqcap C_n$, if value restrictions of the form $\forall r.C_1, \ldots, \forall r.C_n$ exist on the top-level of C; otherwise, $\mathsf{val}_r(C) := \top$;
- $\mathsf{ex}_r(C) := \{C' \mid \text{there exists } \exists r.C' \text{ on the top-level of } C\}$;
- $\mathsf{min}_r(C) := max\{k \mid C \sqsubseteq (\geq k\ r)\}$ (Note that $\mathsf{min}_r(C)$ is always finite.);
- $\mathsf{max}_r(C) := min\{k \mid C \sqsubseteq (\leq k\ r)\}$; if there exists no k with $C \sqsubseteq (\leq k\ r)$, then $\mathsf{max}_r(C) := \infty$.

Now, an $\mathcal{ALCN}$-concept description C in $\mathcal{ALCN}$-NF is of the form $C = C_1 \sqcup \ldots \sqcup C_n$ with $C_i :=$

$$\bigsqcap_{A \in prim(C_i)} A \sqcap \bigsqcap_{r \in N_R} \Big(\bigsqcap_{C' \in \mathsf{ex}_r(C_i)} \exists r.C' \sqcap \forall r.\mathsf{val}_r(C_i) \sqcap (\geq \mathsf{min}_r(C_i)\ r) \sqcap (\leq \mathsf{max}_r(C_i)\ r)\Big),$$

for all $i = 1, \ldots n$, where the concept descriptions $\mathsf{val}_r(C_i)$ and C' again are in $\mathcal{ALCN}$-normal form and C_i is removed from the disjunction in case $C_i \equiv \bot$.

Next, implicit information captured in the concept description is made explicit. Due to space limitations, we can only give an intuition in which combinations of concept constructors information is induced. For a thorough discussion refer to [6] or [2]. In case of the number restrictions appearing in the concept description $(\geq \mathsf{min}_r(C)\ r)$ and $(\leq \mathsf{max}_r(C)\ r)$ already make induced information explicit. At-least restrictions can be induced by incompatible existential restrictions, e.g., $\exists r.A \sqcap \exists r.\neg A$ induces $(\geq 2\ r)$. At-most restrictions can be induced only by value restrictions equivalent to $\forall r.\bot$ implying $(\leq 0\ r)$. Vice versa, value restrictions can be induced by $(\leq 0\ r)$. Furthermore, value restrictions can be implied, if the minimal number of r-successors required by either at-least or by incompatible existential restrictions coincide with $\mathsf{max}_r(C)$. For example in $(\leq 2\ r) \sqcap (\exists r.A \sqcap B) \sqcap (\exists r.A \sqcap \neg B)$ the value restriction $\forall r.A$ is induced. Induced value restrictions are denoted $\mathsf{ind\text{-}val}_r(C)$.

Input: $\mathcal{ALCN}$-concept description C. **Output:** $\mathcal{ALCN} - \mathcal{ALEN}$-approximation of C.

1. If $C \equiv \bot$ then $\mathsf{c\text{-}approx}_{\mathcal{ALEN}}(C) := \bot$ elseif $C \equiv \top$ then $\mathsf{c\text{-}approx}_{\mathcal{ALEN}}(C) := \top$.
2. Otherwise, transform C into $\mathcal{ALCN}$-normal form $C_1 \sqcup \cdots \sqcup C_n$ and return

$$\begin{aligned}
&\mathsf{c\text{-}approx}_{\mathcal{ALEN}}(C) := \\
&\sqcap_{A \in \bigcap_i prim(C_i)} A \\
&\sqcap\ (\geq \min\{\mathsf{min}_r(C_i) \mid 1 \leq i \leq n\}\ r)\ \sqcap\ (\leq \max\{\mathsf{max}_r(C_i) \mid 1 \leq i \leq n\}\ r) \\
&\sqcap \prod_{\substack{(C_1',\ldots,C_n') \in \\ \mathsf{ind\text{-}ex}_r(C_1) \times \cdots \times \mathsf{ind\text{-}ex}_r(C_n)}} \exists r.\mathsf{lcs}\{\mathsf{c\text{-}approx}_{\mathcal{ALEN}}(C_i' \sqcap \mathsf{val}_r(C_i)) \mid 1 \leq i \leq n\} \\
&\sqcap \forall r.\mathsf{lcs}\{\mathsf{c\text{-}approx}_{\mathcal{ALEN}}(\mathsf{ind\text{-}val}_r(C_i)) \mid 1 \leq i \leq n\}
\end{aligned}$$

Fig. 1. The computation algorithm for $\mathcal{ALCN}$-$\mathcal{ALEN}$-approximation.

Induced existential restrictions are obtained if $||\mathsf{ex}_r(C)|| > \mathsf{max}_r(C)$. In this case the existential restrictions have to be merged. More precisely, such a merging must yield $\mathsf{max}_r(C)$ existential restrictions s.t. the set $\mathsf{ex}_r(C)$ is partitioned and only consistent existential restrictions are obtained. Moreover, $\mathsf{val}_r(C)$ has to be propagated onto each existential restriction. The induced existential restrictions are obtained by computing the commonalities of all ways of obtaining valid mergings.

Figure 1 displays the computation algorithm for $\mathcal{ALCN}$-$\mathcal{ALEN}$-approximation. In the computation of the approximation as well as in the computation of $\mathsf{ind\text{-}val}_r(C)$ and $\mathsf{ind\text{-}ex}_r(C)$ the lcs for $\mathcal{ALEN}$ is used, which was introduced in [6].

3 Nice concepts for Approximation

In general, the computation of an approximation cannot be split at the conjunction because of possible interactions—in case of $\mathcal{ALC}$-$\mathcal{ALE}$-approximation between existential and value restrictions on the one hand and inconsistencies induced by negation on the other. For example, the approximation $\mathsf{c\text{-}approx}_{\mathcal{ALE}}(\exists r.\top \sqcap (\forall r.A \sqcup \exists r.A))$ yields $\exists r.A$ while the conjunct-wise version $\mathsf{c\text{-}approx}_{\mathcal{ALE}}(\exists r.\top) \sqcap \mathsf{c\text{-}approx}_{\mathcal{ALE}}(\forall r.A \sqcup \exists r.A)$ only produces $\exists r.\top$. In [4] those concept descriptions were called *nice* for which this strategy still produces the correct result.

Definition 3 (nice concepts). *Let $C := C_1 \sqcap \cdots \sqcap C_n$ be a $\mathcal{L}_1$-concept description. C is nice if $\mathsf{approx}_{\mathcal{L}_2}(C) \equiv \mathsf{approx}_{\mathcal{L}_2}(C_1) \sqcap \cdots \sqcap \mathsf{approx}_{\mathcal{L}_2}(C_n)$.*

For these concept descriptions interactions between conjuncts are excluded. Since the use of nice concept descriptions is to speed-up approximation, it is important that the conditions to distinguish these concept descriptions can be tested easily. Therefore the test for nice concept descriptions should be based on simple discrimination conditions. To this end the conditions for nice $\mathcal{ALC}$-concept descriptions given in [4] are sound, but not complete:

1. The restrictions are limited to one type per role-depth: on every role depth of a nice concept either no $\forall$-restrictions or no $\exists$-restrictions occur.

2. A concept name and its negation may not occur on the same role-depth of a nice concept.

It is shown in [4] that for $\mathcal{ALC}$-concept descriptions fulfilling these conditions conjunct-wise approximation is correct. The above conditions are intuitive and easy to test, but very strict. Consider the concept description $(\exists r.\exists s.A \sqcup B) \sqcap (\exists t.\forall s.\neg A \sqcup C)$, which violates both conditions. However, we get the correct result, if the conjuncts are approximated independently, since the relevant concept descriptions are nested in existential restrictions for different roles. In general, a concept description can still be approximated conjunct-wise, if the "interacting" concept descriptions are reachable via different role paths. Too strict conditions to distinguish nice concept descriptions would rule out too many concept descriptions that could actually be approximated conjunct-wise. In these cases the "expensive" approximation must be applied. In order to be able to classify more concept descriptions as nice, we devise relaxed conditions for nice concept descriptions.

3.1 Nice $\mathcal{ALCN}$-concept descriptions

The conditions for nice $\mathcal{ALCN}$-concept descriptions have to take into account the information induced by number restrictions in combination with other concept constructors, i.e., the interactions we sketched in Section 2.1. For example, consider the $\mathcal{ALCN}$-concept description $C = C_1 \sqcap C_2$, where the conjuncts are: $C_1 = (\exists s.A) \sqcap (\exists s.\neg A) \sqcup \bot$ and $C_2 = (\leq 1\ s) \sqcup B$. Now, if we approximate C conjunct-wise we obtain $\mathsf{c\text{-}approx}_{\mathcal{ALEN}}(C_1) = (\exists s.A) \sqcap (\exists s.\neg A)$ and $\mathsf{c\text{-}approx}_{\mathcal{ALEN}}(C_2) = \top$, yielding $(\exists s.A) \sqcap (\exists s.\neg A) \sqcap \top$ as the combined result. Due to the incompatibilities between C_1 and C_2 for the information on the role s, the disjunction in C_2 collapses to B. The approximation yields $\mathsf{c\text{-}approx}_{\mathcal{ALEN}}(C) = (\exists s.A) \sqcap (\exists s.\neg A) \sqcap B$, which is more specific than the conjunct-wise approximation.

To devise syntactic conditions to detect nice $\mathcal{ALCN}$-concept descriptions, we introduce notation to access the numbers in number restrictions. For an $\mathcal{ALCN}$-concept description C and a role r let $\mathsf{at\text{-}least}_r(C)$ ($\mathsf{at\text{-}most}_r(C)$) denote the maximal (minimal) number appearing in at-least (at-most) restrictions on the top-level of C or, if the top-level of C has no at-least restriction, 0 (at-most restriction, ∞.) Next, we specify the notion of sub-concept descriptions accessible by a role path, that will be used in the conditions for nice concept descriptions.

Definition 4. *Let a* Qr-path *(denoted ρ) be defined as $\rho = (Q\,r)^*$ for $Q \in \{\exists, \forall\}$ and $r \in N_R$ and let λ denote the empty Qr-path. Let C be an $\mathcal{ALCN}$-concept description and ρ be a Qr-path, then $sub(C, \rho)$:=*

- $prim(C) \sqcap \bigsqcap_{s \in N_R} (\leq \mathsf{at\text{-}most}_s(C)\ s) \sqcap (\geq \mathsf{at\text{-}least}_s(C)\ s)$, *if* $\rho = \lambda$,
- $sub(\mathsf{val}_r(C), \rho')$, *if* $\rho = \forall\, r \cdot \rho'$ *and* $\mathsf{val}_r(C) \not\equiv \top$,
- $\bigcup_{C' \in \mathsf{ex}_r(C)} sub(C', \rho')$, *if* $\rho = \exists\, r \cdot \rho'$,
- $\emptyset$, *otherwise.*

Based on role-paths we can give sufficient conditions for nice $\mathcal{ALCN}$-concept descriptions, which additionally relax the too strict conditions given in [4] for nice $\mathcal{ALC}$-concept descriptions.

Definition 5 (suffcient conditions for nice $\mathcal{ALCN}$-concept descriptions). *Let C be an $\mathcal{ALCN}$-concept description in NNF. Then C is* nice*, if for every Qr-path ρ with $C_1, C_2 \in sub(C, \rho)$ and C_1', C_2' denoting C_1, C_2 in $\mathcal{ALCN}$-NF and all $r \in N_R$ it holds*

1. $||\{\exists \mid \mathsf{at\text{-}least}_r(C_i) \neq 0, i \in \{1,2\}\} \cup \{\exists \mid \mathsf{ex}_r(C_1) \cup \mathsf{ex}_r(C_2) \neq \emptyset\}|| \; + \; ||\{\forall \mid \mathsf{at\text{-}most}_r(C_i) \neq \infty, i \in \{1,2\}\} \cup \{\forall \mid \bigsqcap_{i\in\{1,2\}} \mathsf{val}_r(C_i) \not\equiv \top\}|| \; \leq 1$, *and*
2. $prim(C_1) \cup prim(C_2)$ *does not contain a concept name and its negation.*

The Condition 1 rules out three constellations for sub-concept description accessible via the same Qr-path: (1) concept descriptions that induce role successors (either by existential or by at-least restrictions) and that have a possibly induced value restriction and (2) concept descriptions with contradicting number restrictions and (3) concept descriptions that require merging of existential restrictions. Condition 2 rules out concept descriptions that have a concept name and its negation accessible via the same Qr-path. To show that concept descriptions fulfilling the conditions from Definition 5 can be approximated conjunct-wise, we adapt the proof from [4] in [7]. The following lemma is central in the proof of the correctness of the conditions for nice concepts.

Lemma 1. *For $1 \leq i \leq 2$, let C_i and D_i be $\mathcal{ALEN}$-concept descriptions such that $C_1 \sqcap C_2 \sqcap D_1 \sqcap D_2$ fulfill all conditions from Definition 5. Then it holds that $\mathsf{lcs}\{C_i \sqcap D_j \mid i,j \in \{1,2\}\} \equiv \mathsf{lcs}\{C_1, C_2\} \sqcap \mathsf{lcs}\{D_1, D_2\}$.*

The following theorem states our claim that approximations of $\mathcal{ALCN}$-concept descriptions fulfilling the conditions from Definition 5, can be obtained by a conjunction of approximations of the conjuncts from the original concept.

Theorem 1. *Let $C \sqcap D$ be a nice $\mathcal{ALCN}$-concept description, then $\mathsf{c\text{-}approx}_{\mathcal{ALEN}}(C \sqcap D) \equiv \mathsf{c\text{-}approx}_{\mathcal{ALEN}}(C) \sqcap \mathsf{c\text{-}approx}_{\mathcal{ALEN}}(D)$.*

The claim is proved by induction over the sum of the nesting depths of $\sqcap$ and $\sqcup$ on every role level in C and D. For the induction step a case distinction is made depending on whether C or D are conjunctions or disjunctions. If at least one concept description is a disjunction the approximation is defined as the lcs of all $\mathcal{ALCN}$-normalized and approximated disjuncts (if one of the concept descriptions is a conjunction, it firstly has to be distributed over the disjunction). The main idea then is to use Lemma 1 to transform single lcs calls of a certain form into a conjunction of lcs calls which eventually leads to the conjunction of the approximations of C and D. The full proof can be found in [7].

Now, the actual $\mathcal{ALCN}$-$\mathcal{ALEN}$-approximation algorithm can be adapted by first testing whether a concept description is nice and then performing conjunct-wise approximation for each conjunct and conjoin the results. The algorithm performs the nice test at the beginning of every recursive call, unless the current (sub-)concept description is already known to be nice.

4 Implementation

The implementation of approximation for nice $\mathcal{ALCN}$-concept descriptions requires little changes in the implementation of the approximation algorithm. The major part to implement is the tester for nice concept descriptions `nice-p` . Since this test has to be performed at the beginning of every approximation computation, the implementation must be very efficient.

The procedure `nice-p` in our implementation realizes the necessary conditions for nice $\mathcal{ALCN}$-concept descriptions from Definition 5. Our implementation of the nice test employs a couple of optimizations. Firstly, some steps are only taken on demand, such as unfolding and the transformation into NNF. Furthermore, `nice-p` stores information of a certain named concept, say C, in a so-called info-table, where the key of this info-table is a path ρ and the value is the sub(C,ρ). This enhances the checking of the conditions from Definition 5 "on-the-fly". Suppose that while unfolding and transforming concept C, we encounter the concept name A inspecting Qr-path ρ. Then, we update the info-table of C by adding the concept name A to the value of the key ρ. Before we add A, we first check whether adding A violates Condition 2 in Definition 5. A similar procedure is carried out, if we encounter number, value and existential restrictions during the unfolding and transformation. Furthermore, our implementation does not only use dynamic programming to re-use results obtained during the current computation on whether a concept is already known to be nice or not, but caches these results. Based on this cache the already obtained information on whether a named concept is nice or not is re-used in subsequent runs of `nice-p` .

We extended the implementation of $\mathcal{ALCN}$-$\mathcal{ALEN}$-approximation from our non-standard inference system Sonic [8] to the use of nice $\mathcal{ALCN}$-concepts in a naive way. This way the full potential of the conjunct-wise application of concepts is not yet used. One can in addition implement a caching strategy based on nice concepts in the following way: if we want to approximate a conjunction that is nice and we have computed the approximation of some of its conjuncts already, we only need to conjoin the cached values with the freshly computed approximations of the remaining conjuncts. The implementation of this technique is future work.

5 First tests

Although our implementation is not yet mature, we can already report on some experiences. Most importantly, it was unknown whether nice concepts do appear in TBoxes from real-world applications and if, how frequently. We tested our implementation of `nice-p` on the DICE TBox, which is a medical knowledge base from the intensive care domain, see [5]. The DICE TBox contains about 3500 concepts, of which 3249 have a (primitive) definition, and DICE is an unfoldable TBox. Originally, this TBox uses $\mathcal{ALCQ}$ (and disjointness statements), we used a variant of it pruned down to $\mathcal{ALCN}$ for our tests. It turned out that this knowledge base has 493 nice concepts satisfying the necessary conditions from

Definition 5. These are concepts not only with a nice sub-concept description, but which are nice "completely". So a first result of our test is that nice $\mathcal{ALCN}$ concepts do appear in knowledge bases from practical applications. In case of the DICE knowledge base about 14.3% of all named concepts with a definition are nice. This might not seem very much at first, but indicates that for many concepts significant parts of the unfolded concept descriptions are nice and can be approximated independently. The run-time we measured for each call of `nice-p` when testing all concepts in DICE, was $0.91s$ on the average. This run-time includes the time needed for unfolding and transforming the concept into NNF—steps that the approximation algorithm requires anyway.

We tested our implementation of conjunct-wise $\mathcal{ALCN}$-$\mathcal{ALEN}$-approximation on those 463 concepts from the DICE knowledge base that are nice. It turned out that the implementation of the ordinary approximation algorithm needed $1.89s$ per concept on the average, while the nice approximation needed $1.44s$. This is a speed-up of about 24%. Taking into account that the time for nice approximation included the time for the `nice-p` test, one can say that it performed reasonably well.

In this paper we have extended necessary conditions for nice concepts to $\mathcal{ALCN}$-concepts. The notion of nice concepts is the basis for the two optimization techniques for concept approximation: non-naive caching and conjunct-wise approximation. A first test of our implementation of nice concepts showed that these kind of concepts do appear in knowledge bases from practical application and that conjunct-wise approximation results in a substantial performance gain.

References

1. F. Baader and W. Nutt. *The Description Logic Handbook: Theory, Implementation, and Applications*, chapter Basic Description Logics. Cambr. Univ. Press, 2003.
2. S. Brandt, R. Küsters, and A.-Y. Turhan. Approximating $\mathcal{ALCN}$-concept descriptions. In *Proc. of the 2002 Description Logic Workshop (DL 2002)*, 2002.
3. S. Brandt, R. Küsters, and A.-Y. Turhan. Approximation and difference in description logics. In *Proc. of the 8th Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR-02)*. Morgan Kaufm. Publ., 2002.
4. S. Brandt and A.-Y. Turhan. An approach for optimized approximation. In *Proc. of the 2002 Applications of Description Logic Workshop (ADL 2002)*, 2002. LTCS-Report 02-03 is an extended version, see `http://lat.inf.tu-dresden.de/research/reports.html`.
5. R. Cornet and A. Abu-Hanna. Using description logics for managing medical terminologies. In *A. I. in Medicine: 9th Conference on Artificial Intelligence in Medicine in Europe (AIME 2003)*. Springer, 2003.
6. R. Küsters and R. Molitor. Computing Least Common Subsumers in $\mathcal{ALEN}$. In B. Nebel, editor, *Proc. of the 17th Int. Joint Conf. on Artificial Intelligence (IJCAI-01)*. Morgan Kaufman, 2001.
7. A.-Y. Turhan. *On the computation of common subsumers*. PhD thesis, TU Dresden, Institute for Theoretical Computer Science, 2007. forthcoming.
8. A.-Y. Turhan and C. Kissig. Sonic — Non-standard inferences go OilEd. In *Proc. of the Int. Joint Conf. on Automated Reasoning (IJCAR-04)*. Springer, 2004.