Anytime Classification by Ontology Approximation

S.Schlobach, E.Blaauw, M.El Kebir, A.ten Teije, F.van Harmelen, S.Bortoli, M.Hobbelman, K.Millian, Y.Ren, S.Stam,, P.Thomassen, , R.van het Schip, W.van Willigem

Vrije Universiteit Amsterdam, contact:schlobac@few.vu.nl

.

Abstract. Reasoning with large or complex ontologies is one of the bottle-necks of the Semantic Web. In this paper we present an anytime algorithm for classification based on approximate subsumption. We give the formal definitions for approximate subsumption, and show its monotonicity and soundness; we show how it can be computed in terms of classical subsumption; and we study the computational behaviour of the algorithm on a set of realistic benchmarks. The most interesting finding is that anytime classification works best on ontologies where classical subsumption is hardest to compute.

1 Introduction

Reasoning with large or complex terminology is computationally difficult and is one of the known bottlenecks in application of ontologies on the Semantic Web. It is known that for expressive ontology languages such as OWL, most reasoning services are untractable, i.e. already for very small ontologies sound and complete reasoning is practically infeasible. Also for some applications it is more important to have some answers quickly, rather than all answers after a long period of waiting. In both cases, approximate reasoning can be useful, in particular when algorithms are monotonic: when with increasing time the quality of the output increases.

Cadoli and Schaerf proposed a formal method of approximation of propositional entailment in their seminal paper [1], which also includes some ideas on approximation of Description Logics reasoning. Stuckenschmidt [2] recently expanded these ideas to ALCQ concept subsumption, and provided a simple algorithm so that approximate subsumption can be calculated using standard DL reasoner. The idea is, basically, to ignore a subset of the atomic concepts in the query (i.e. interpret them as trivially fulfilled, namely as always true), or alternatively to interpret the literals for these atoms as always false.

In this paper we will extend this idea to approximate terminological subsumption, i.e. the question whether two concepts C and D subsume each other in the presence of a terminology $\mathcal{T}$. Instead of the approaches proposed in [2, 3] which answers an approximated query Q' based on an ontology O (i.e: $O \vdash Q'$?)

we approximate the ontology O rather than the query Q (i.e: $O' \vdash Q$?). As reasoning with ontologies is computationally much harder than purely conceptual reasoning, the need for approximation seems even more urgent in this case. The proposed method (more details later) is sound, but incomplete. We also show that the sequence of approximations is monotonic, i.e. that with decreasing set of ignored concepts, none of the logical consequences is lost. This result is the basis for an *anytime algorithm* for ontology reasoning.

The variable factor in the anytime algorithm is then the choice of the set of concepts that remain intact, in Cadoli and Schaerf's work called the set $\mathcal{S}$. For an anytime algorithm we need to produce a sequence $\mathcal{S}$ of subsets $\varnothing \subseteq S_0 \subseteq \ldots S_n \subseteq S$, where S is the set of all atoms in the TBox and query, such that most of the consequences are found as early as possible. Examples for possible choices could be purely random orders, frequency or connectedness, but also the use of other ontologies as background knowledge is conceivable.

In this paper we evaluate the quality of such an anytime algorithm for the problem of *classification of atomic concept names*, comparing both the result of the classification hierarchy calculated for an approximated terminology, as well as the runtime needed for classification.

2 Approximating terminological reasoning

In their seminal paper [1] Cadoli and Schaerf introduced approximate reasoning for Propositional as well as Description Logics, and our approach is a rather simple combination of their two results. The main idea is to treat only a particular subset S of the vocabulary[1] in the classical way, and either ignore the rest, or interpret the remaining concepts as necessarily empty. The hope is that such an S-approximated ontology reasoning becomes computationally easier, but where still some of the logical consequences remain valid. In existing work on approximation of DL reasoning, usually the query is approximated. In our approach, we approximate the terminology, i.e. we consider a different set of models for a TBox, and calculate logical consequences for these models in the standard way. We will first recall the basic notions of DL reasoning before outlining our proposed methods in all detail.

2.1 Classical subsumption

For Description Logics the handbook [4] is an excellent reference. In the classical definition of a model of a terminology (TBox) in Description Logics, $\mathcal{U}$ is a set of objects, called the universe, and $(\cdot)^{\mathcal{I}}$ a mapping, which interprets DL concepts as subsets of $\mathcal{U}$. An interpretation $\mathcal{I} = (U, (\cdot)^{\mathcal{I}})$ is then called a *model* of a terminological axiom $C \sqsubseteq D$, if, and only if, $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$. A *model for a TBox* $\mathcal{T}$ is an interpretation which is a model for all axioms in $\mathcal{T}$. Based on these semantics

[1] In this paper we will only consider concept names, but extension to relations is conceptually straightforward.

the notion of subsumption with respect to $\mathcal{T}$ is defined formally: a concept D is subsumed by a concept C in a terminology $\mathcal{T}$ (we write $\mathcal{T} \models C \sqsubseteq D$) if, and only if, $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ for all models $\mathcal{I}$ of $\mathcal{T}$.

This definition is based on the classical interpretation of a concept, for example, in $\mathcal{ALC}$, which is a simple yet relatively expressive DL with conjunction ($C \sqcap D$), disjunction ($C \sqcup D$), negation ($\neg C$) and universal ($\forall r.C$) and existential quantification ($\exists r.C$).[2] The interpretation function interprets an atomic concept A as a subset $A^{\mathcal{I}}$ of the domain, and is extended to the different language constructs as follows:

- $(C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}}$
- $(C \sqcup D)^{\mathcal{I}} = C^{\mathcal{I}} \cup D^{\mathcal{I}}$
- $(\neg C)^{\mathcal{I}} = \mathcal{U} \setminus C^{\mathcal{I}}$
- $(\exists R.C)^{\mathcal{I}} = \{d \in \mathcal{U} \mid \exists e \in \mathcal{U} : (d, e) \in R^{\mathcal{I}} \text{ and } e \in C^{\mathcal{I}}\}$
- $(\forall R.C)^{\mathcal{I}} = \{d \in \mathcal{U} \mid \forall e \in \mathcal{U} : (d, e) \in R^{\mathcal{I}} \text{ implies } e \in C^{\mathcal{I}}\}$

2.2 Approximate terminological subsumption

The idea of approximate terminological reasoning is now based on interpreting an ontology in a non-standard way. In particular, the idea is to look at particular subsets and supersets of interpretations called *lower* and *upper approximations* of an interpretation. An approximate model is then an interpretation which interprets any axiom in the an approximate way. Following the ideas of Cadoli & Schaerf [1] we restrict the vocabulary that is taken into account in the definition of an interpretation to a set usually denoted by the letter S. We will call S the *approximation set*. In the upper approximation, those atoms not in S are usually interpreted as empty sets, in the lower one as the entire domain.

The following definition of lower and upper S-approximation follow closely the work of Cadoli and Schaerf [1]:

Definition 1 (Lower and Upper Approximation). *The lower approximation* $(.)^{\mathcal{I}_S^-}$ *and the upper approximation* $(.)^{\mathcal{I}_S^+}$ *are defined as follows:*

– *for atomic concepts* A*,*

$$\begin{aligned} A^{\mathcal{I}_S^-} &= A^{\mathcal{I}_S^+} = A^{\mathcal{I}} && \text{if } A \in S \\ A^{\mathcal{I}_S^-} &= \varnothing \text{ and } A^{\mathcal{I}_S^+} = \mathcal{U} && \text{if } A \notin S \end{aligned}$$

– *for the Boolean operators:*

$$\begin{aligned} (\neg C)^{\mathcal{I}_S^-} &= \mathcal{U} \setminus C^{\mathcal{I}_S^+} & \qquad (\neg C)^{\mathcal{I}_S^+} &= \mathcal{U} \setminus C^{\mathcal{I}_S^-} \\ (C \sqcap D)^{\mathcal{I}_S^-} &= C^{\mathcal{I}_S^-} \cap D^{\mathcal{I}_S^-} & (C \sqcap D)^{\mathcal{I}_S^+} &= C^{\mathcal{I}_S^+} \cap D^{\mathcal{I}_S^+} \\ (C \sqcup D)^{\mathcal{I}_S^-} &= C^{\mathcal{I}_S^-} \cup D^{\mathcal{I}_S^-} & (C \sqcup D)^{\mathcal{I}_S^+} &= C^{\mathcal{I}_S^+} \cup D^{\mathcal{I}_S^+} \end{aligned}$$

[2] Most definitions can be extended to more expressive languages, but throughout the paper we will work with $\mathcal{ALC}$ for the sake of simplicity.

– *for the quantifiers:*

$$
\begin{aligned}
(\exists R.C)^{\mathcal{I}_S^-} &= \{d \in \mathcal{U} \mid \exists e \in \mathcal{U} : (d,e) \in R^{\mathcal{I}} \text{ and } e \in C^{\mathcal{I}_S^-}\} \\
(\forall R.C)^{\mathcal{I}_S^-} &= \{d \in \mathcal{U} \mid \forall e \in \mathcal{U} : (d,e) \in R^{\mathcal{I}} \text{ implies } e \in C^{\mathcal{I}_S^-}\} \\
(\exists R.C)^{\mathcal{I}_S^+} &= \{d \in \mathcal{U} \mid \exists e \in \mathcal{U} : (d,e) \in R^{\mathcal{I}} \text{ and } e \in C^{\mathcal{I}_S^+}\} \\
(\forall R.C)^{\mathcal{I}_S^+} &= \{d \in \mathcal{U} \mid \forall e \in \mathcal{U} : (d,e) \in R^{\mathcal{I}} \text{ implies } e \in C^{\mathcal{I}_S^+}\}
\end{aligned}
$$

Stuckenschmidt [2] shows the property of monotonicity $C^{\mathcal{I}_S^-} \subseteq C^{\mathcal{I}} \subseteq C^{\mathcal{I}_S^+}$ for any subset S of the concept-names. The property of generalised monotonicity shown by Stuckenschmidt seems incorrect, it follows a corrected version (with the beginning of a proof in the appendix).

Lemma 1 (Generalised Monotonicity) *Given a lower $\mathcal{I}_S^-$ and an upper S-approximate interpretation $\mathcal{I}_S^+$, and two sub-vocabularies $S_1 \subseteq S_2$, the following equations hold for all concept expressions C.*

$$1)\ C^{\mathcal{I}_{S_1}^-} \subseteq C^{\mathcal{I}_{S_2}^-} \qquad 2)\ C^{\mathcal{I}_{S_2}^+} \subseteq C^{\mathcal{I}_{S_1}^+}$$

Intuitively Lemma 1 states that with increasing size of the approximation set S the size of the interpretation of the concepts increases for $\mathcal{I}_S^-$ and decreases for $\mathcal{I}_S^+$.

Based on these approximations of an interpretation we can now define the notion of an approximate model for a terminology. The basic intuition is that we slightly release the constraints on the terminology by forcing the left-hand sides (lhs) of axioms to be more specific, and the right-hand side (rhs) of the axioms to be more general, than in the classical case. This can be achieved by considering also interpretations to be models of an axioms, in which the lower approximation of the lhs is a subset of the upper approximation of the rhs.

Definition 2 (S-approximate models). *Let C and D be Description Logic concepts, and $\mathcal{T}$ a TBox. An interpretation $\mathcal{I}$ is an S-*approximate model of an axiom $C \sqsubseteq D$ *if, and only if, $C^{\mathcal{I}_S^-} \sqsubseteq D^{\mathcal{I}_S^+}$. An interpretation $\mathcal{I}$ is an S-*approximate model of the TBox $\mathcal{T}$ *if, and only if, it is an S-approximate model of all axioms $ax \in \mathcal{T}$. In this case we write $\mathcal{I} \models_S \mathcal{T}$.*

Approximations preserve models, i.e. if an interpretation is a model for an axiom or a TBox, it is also an S-approximate model for any subset S of the vocabulary.

Lemma 2 *If $\mathcal{I}$ is a (classical) model for an axioms ax, it is an S-approximate model for ax for any S. Similarly, if $\mathcal{I}$ is a classical model for a TBox $\mathcal{T}$, it is an S-approximate model for $\mathcal{T}$.*

Finally we need to consider monotonicity for increasing size of set S to be able to apply an anytime algorithm to approximate classification.

Lemma 3 *Let $S_1 \subseteq S_2$, $\mathcal{T}$ be a TBox. Then, $\mathcal{I} \models_{S_2} \mathcal{T}$ implies that $\mathcal{I} \models_{S_1} \mathcal{T}$.*

Note that this lemma shows inverse monotonicity for models, i.e. with increasing size of S there are fewer models for any TBox.

Approximate subsumption is now defined in the classical way as a subset relation between interpreted concepts, but now we only take into account the *approximate models* of a TBox as previously defined (rather than the classical models).

Definition 3 (Approximate subsumption). *Let $\mathcal{T}$ be a TBox. We say that a concepts D S-*approximately subsumes *a concept C w.r.t. a TBox $\mathcal{T}$ (abbreviated $\mathcal{T} \models_S C \sqsubseteq D$) if, and only if, $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ for all S-approximate models $\mathcal{I}$ of $\mathcal{T}$.*

Note that this definition is crucially different to previous approaches to approximated subsumption: e.g, in [2], the subset relation is interpreted non-classically; in our case, we consider a different set of models.[3]

From this definition and the monotonicity of terminological reasoning monotonicity for approximated subsumption follows.

Theorem 1 (Monotonocity). *Let $S_1 \subseteq S_2$ and $\mathcal{T}$ be a TBox, and C and D DL concepts. We then have that $\mathcal{T} \models_{S_1} C \sqsubseteq D$ implies that $\mathcal{T} \models_{S_2} C \sqsubseteq D$.*

The theorem follows immediately from Lemma 3. This theorem also immediately implies the soundness of approximate subsumption (simply take in the theorem for S_2 the set of all atoms in $\mathcal{T}$):

Corrolary 1 (Soundness) *For any S_i, $\mathcal{T} \models_{S_i} C \sqsubseteq D$ implies $\mathcal{T} \models C \sqsubseteq D$*

A note of caution seems in order: in Theorem 1 we only consider approximations of the terminology, and the query is interpreted classically. We can also approximate the queries, but in this paper we restrict our attention to non-approximate queries as, in this paper, we are interested in approximate classification of concept names only.

2.3 Calculating approximate subsumption

Now, what does this all mean? Based on theorem 1 we can calculate an approximation of the classical subsumption hierarchy (classification) in an incremental, i.e. anytime, way. The idea is simple, starting with a small S we can calculate the classification step by step by approximating the classical interpretation of $\mathcal{T}$. Theorem 1 guaranties that any subsumption relation we find, is also valid in the un-approximated TBox. As we will show that we can effectively reduce S-approximate subsumption to classical reasoning (thus being able to use standard reasoner), and as S-approximations are easier (and thus faster) to calculate, we have developed a new way to iteratively create the classical subsumption hierarchy, which is guarantied to find the correct solution at the end of the iteration.

[3] Of course, combinations of both approaches are conceivable. As in this paper, we only consider the taxonomy between atomic concepts, approximation of the query is necessarily trivial.

What remains to show is that there is an efficient way to reduce S approximated subsumption (and inner S approximated subsumption) to classical reasoning. The basic idea follows again the proposal of Stuckenschmidt in [2], and is a simple recursive rewriting procedure in which the atoms outside S are replaced by either $\bot$ or $\top$. Stuckenschmidt defines two procedures $(\cdot)^{+^S}$ and $(\cdot)^{-^S}$ in a recursive way, where the left-hand side of an equation is rewritten by the right-hand side. A slightly adapted version is given in the following definition:

Definition 4 (Rewrite Procedure). *The rewriting procedures* $(\cdot)^{+^S}$ *and* $(\cdot)^{-^S}$ *are defined as follows:*

$$
\begin{array}{rlrll}
A^{-^S} & = A^{\mathcal{I}} & A^{+^S} & = A^{\mathcal{I}} & \text{if } A \in S \\
A^{-^S} & = \bot & A^{+^S} & = \top & \text{if } A \notin S \\
(\neg C)^{-^S} & = \neg C^{+^S} & (\neg C)^{+^S} & = \neg C^{-^S} & \\
(C \sqcap D)^{-^S} & = C^{-^S} \sqcap D^{-^S} & (C \sqcap D)^{+^S} & = C^{+^S} \sqcap D^{+^S} & \\
(C \sqcup D)^{-^S} & = C^{-^S} \sqcup D^{-^S} & (C \sqcup D)^{+^S} & = C^{+^S} \sqcup D^{+^S} & \\
(\exists R.C)^{-^S} & = \exists R.C^{-^S} & (\exists R.C)^{+^S} & = \exists R.C^{+^S} & \\
(\forall R.C)^{-^S} & = \forall R.C^{-^S} & (\forall R.C)^{+^S} & = \forall R.C^{+^S} &
\end{array}
$$

It is easy to see that both procedures terminate on any concept C in $\mathcal{ALC}$ as the complexity of the formula decreases in any application of the rules. If in a TBox $\mathcal{T}$ any axiom $C \sqsubseteq D$ has been rewritten as $C^{-^S} \sqsubseteq D^{+^S}$ we will denote the resulting TBox as $\mathcal{T}^S$. This new TBox $\mathcal{T}^S$ can be used to calculate S-approximate subsumption in terms of classical entailment, as the following theorem shows.

Theorem 2. *Let* $\mathcal{T}$ *be a TBox.* $\mathcal{T} \models_S C \sqsubseteq D$ *if, and only if,* $\mathcal{T}^S \models C \sqsubseteq D$.

Theorem 2 provides the basis for an anytime algorithm for calculating the concept hierarchy of a DL terminology: in each step of this algorithm we add a subset of the vocabulary to get a new approximation set S, and calculate the hierarchy according to the partial order $\mathcal{T} \models_S C \sqsubseteq D$. Following Theorems 1 and 2 for increasing S this hierarchy is always a sub-hierarchy of the original classification of increasing quality.

3 Experiments

In the previous section we have introduced the theoretical foundation of an anytime algorithm for concept classification based on approximate terminological subsumption. However, we have not yet discussed how to determine the approximation set, and have yet to show (or disprove) that our method indeed helps to reason with large and expressive ontologies. To answer these questions, we have done a number of experiments, in which we study different choices of approximation sets, and their effect on accuracy of classification, and the respective runtime.

3.1 Data

For our experiments we used the following well known ontologies: DICE, a medical terminology used for registering diagnoses in Intensive Care units[4], MGED[5], an ontology for microarray experiments, UNSPSC[6], an ontology version of a coding system to classify both products and services, FMA[7], the foundational model of anatomy, and the pizza ontology[8].

All ontologies were simplified from their original OWL versions into corresponding $\mathcal{ALC}$ versions. This of course causes some loss of information, since ALC does not allow transitivity or symmetry of roles, no qualified cardinality constraints, no data-valued roles, among other things. Notwithstanding this loss of information, we believe that our experiments still give a good insight into the properties of approximation. Extending the approach to full OWL is planned for future research.

Table 3.1 summarises some properties of these ontologies, e.g. their respective classification time with RACER Version 1.7.24 as well as the number of occurrences of operators. It also contains information about three ontologies Kpoly5,

	classification time (sec)	#axioms	#∀	#∃	#⊓	#⊔	#¬
DICE	60	4859	3734	5606	1951	784	0
MGED	0.2	792	0	171	12	5	12
UNSPSC	7	19590	0	0	0	0	0
FMA	50	3824	9348	17280	2654	2527	0
PIZZA	0.2	1052	23	148	796	26	796
Kpoly5	4	317	0	202	114	0	163
Kt4p13	5	410	0	289	120	0	224
Kphp5	8	242	0	62	179	0	213

Table 1. Some properties of the ontologies used in our experiments

Kt4p13 and Kphp5, which are taken from the DL benchmark[9]. They have been chosen to investigate particular properties of our method as will be discussed later.

3.2 Experimental setup

What remains to be decided is the choice of an iterative method for building an increasing sequence of approximation sets, and a way of evaluating the results.

[4] `http://kik.amc.uva.nl/dice/home.jsp`
[5] `http://mged.sourceforge.net/ontologies/index.php`
[6] `http://www.unspsc.org/`
[7] `http://sig.biostr.washington.edu/projects/fm/AboutFM.html`
[8] `www.co-ode.org/ontologies/pizza`
[9] `http://dl.kr.org/dl98/comparison/data.html`

Choosing approximation sets We implemented three strategies for choosing new concepts to be added to the approximations set: RANDOM (Choose concepts randomly). MORE (add the most occurring first) or LESS (add the least occurring concepts first). There are of course sophisticated strategies, which we will evaluate in future research.

There are several options for choosing concept names to be added to the approximation set in each step of the anytime algorithm. In a relatively arbitrary ad-hoc decision we decided to increase the subset size after each round by 10%.

Measuring Performance To study the effect of our approximation method, and of the choice of the approximation set, we need appropriate performance measures. When evaluating an approximation algorithm, there are usually two measures that are interesting. Namely the speed of the algorithm, and the recall of the approximation.[10]

In our experiments we only consider the runtime of the approximate classification, i.e. the determination of the taxonomy using the reduced T-Box. If we take reasonably sized ontologies for evaluation, we can calculate the "correct" classification, and can calculate the portion of found and not-yet-found consequences for each set S_i $(0 \leq i \leq n)$. Based on this measure of partial completeness we can evaluate our choice of sequence $\mathcal{S}$. When the correct taxonomy (a.k.a golden standard) is known a priori we can rate the approximated one against it.

We will do this by determining all the subsumption relations in the approximated taxonomy and also in the golden standard. Next, we will express the number of *correct* subsumption relations in the approximated taxonomy as a percentage of the number of subsumption relations in the golden standard. In Figure 1 we consider as an example three taxonomies, one is the golden standard, and the other two are different approximations.

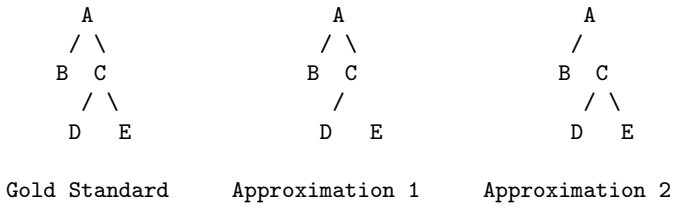

Fig. 1. Example of the recall measure

[10] Note that Theorem 1 ensures that all the found subsumption relations for the approximation must hold in the gold-standard as well. We thus always have 100% precision, and the only interesting measure is the recall.

In the Gold Standard, the following 6 relations hold: $B \sqsubseteq A, C \sqsubseteq A, D \sqsubseteq A, E \sqsubseteq A, D \sqsubseteq C, E \sqsubseteq C$. All of these are also captured in Approximation 1, with the exception of $E \sqsubseteq C$. Approximation 2 however only captures 3 relations ($B \sqsubseteq A, D \sqsubseteq C, E \sqsubseteq C$). This gives Approximation 1 a recall of 5/6=83%, and Approximation 2 a recall of only 3/6=50%.

This performance measure counts the number of subsumption queries that can be answered correctly using the approximated ontology.

The intuition behind this performance metric is that subsumptions higher in the hierarchy are considered more important, since they are involved in establishing more pairwise subsumptions. This performance measure puts a small penalty on mistakes in the details lower down in the ontology, and a high penalty on mistakes in the important top-level categorisations.

3.3 Results

In this section, we will answer the following questions:

1. Under which circumstances is anytime concept classification based on approximate terminological subsumption effective to deal with large and complex ontologies.
2. Does any of the strategies outperform the others, and if so, under which circumstances.

For each of the ontologies described in Section 3.1 we run our experiments 10 times to adjust for statistical outliers in runtime.[11] We then calculate for both runtime and recall the ratio between the value for the gold-standard and the approximation. The desired outcome would be a high recall and low runtime in the early stages of the algorithms, in other words a convex performance profile. We will see that this is often, but not always, achieved.

We will now answer our research questions systematically. In all experiments we run our experiments on all ontologies described in 3.1 and calculated recall and runtime depending on the size of the approximation sets. In all graphics we show runtime and recall in percentage of the gold-standard on the y-axis depending on the size (in percentage of the full vocabulary) of the approximation set on the x-axis. Furthermore, we calculated cumulative runtime for a set S, which is the sum of the run-times for classifying w.r.t. S and all smaller approximation sets previously considered. This measure describes the real gain one makes with an anytime algorithm: as long as the cumulative runtime remains below 100% we gained w.r.t. classical classification. The desired outcome is thus a curve that crosses the 100% mark as late as possible.

Anytime classification benefits some cases Figure 2 shows the results for the three strategies on the DICE terminology, which confirm our hope that anytime classification based on approximate terminological subsumption is a useful reasoning method for large and complex ontologies. In two of the three

[11] Note that, for consistency reasons, we run the experiments for the RANDOM strategy 10 times on the same random sample.

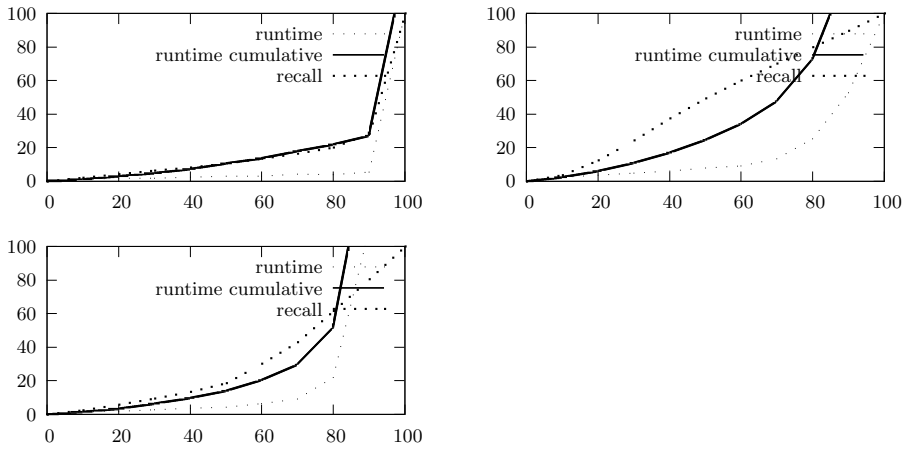

Fig. 2. Results for DICE

methods (MORE and RANDOM), the (relative) recall is significantly higher than the (relative) runtime. Clear winner is the MORE strategy for which the difference betwee recall and runtime is greatest.

Anytime classification does not benefit all cases Unfortunately, not all the results show the same positive properties: in Figure 3 even the best strategy, MORE, never exhibits higher recall than runtime. Basically, this means that relatively more time is lost than correct answers found.

When does anytime classification benefit Given the difference in performance on DICE and UNSPSC, it is now interesting to study in which cases approximate subsumption provides gains over classical subsumption, and when not. For this purpose we compared the differences between relative runtimes and recalls for the remaining ontologies.

The difference between the strategies is shown in the graphs of figure 4. These graphs only show the difference-plots between the percentages of recall and runtime for the three strategies on the different test-cases. At at the y=0 line, the percentage-gain in runtime is off-set by an equally sized percentage-loss in recall, hence no benefit is gained by approximate subsumption. Above this line, the gain in runtime is greater than the loss in recall, hence approximate subsumption creates a benefit, and vice versa below the y=0 line[12]. For example, in the first plot in figure 4, for the succesfull DICE case, it is clearly visible that the MORE strategy always beats classical subsumption because it lies above the y=0 line almost everywhere.

[12] Note that for visual clarity, the 0% baseline (at which the gain in runtime equals the loss in recall) is shifted along the y-axis in the different plots.

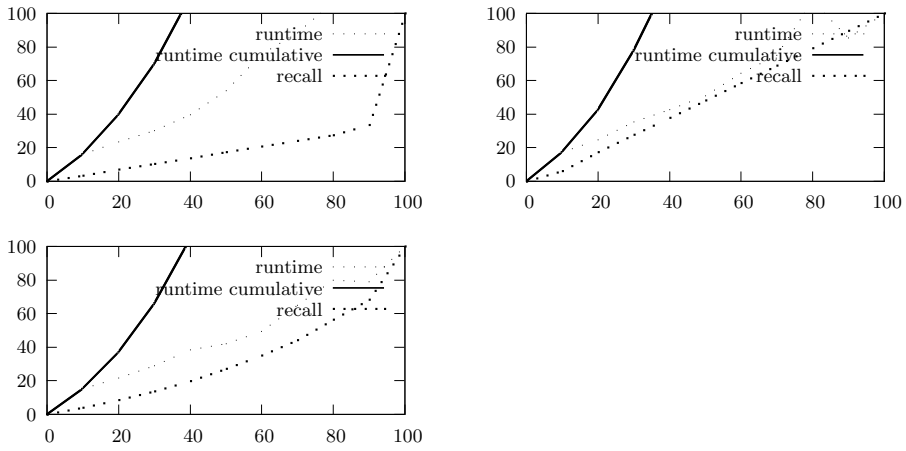

Fig. 3. Results for UNSPSC

The plots in figure 4 show that anytime subsumption beats the classical subsumption for the test-cases DICE, FMA, and the three DL benchmark cases (KPHP5, KT4P13, KPOLY5). For these 5 cases (out of 8), there is at least one strategy that lies above the y=0 trade-off line for nearly all iterations of the algorithm. For the PIZZA case, one strategy (LESS) more or less hovers around the trade-off line without every convincingly beating classical subsumption. For the remaining cases (MGED, FMA), the costs of approximate subsumption are consistently higher than the gains.

The above findings can be explained by a look at the expressiveness of the ontologies in Table 3.1: both DICE and FMA are expressive, and have comparably high classification time, whereas MGED and UNSPSC have almost no expressiveness. The pizza ontology is a bit more expressive, but still easily classifiable. This seems to indicate that our method is most suitable for expressive ontologies that are difficult to classify.

This is of course very good news: *Approximate subsumption works best in exactly those cases when it is needed most*, namely for those ontologies that are expensive to classify with classical algorithms.

Furthermore, it seems that absolute runtime costs are not the right indicator: the classification time of UNSPSC in table 3.1 is comparable to those of the three DL-benchmark ontologies. Still, UNSPSC should be considered "easy" in classical terms, because it contains two orders of more axioms than the DL-benchmark ontologies.

Which strategy works best? What remains to be investigated is which strategy performs best. When studying the plots in figure 4 for those 5 out of 8 cases wheren approximate subsumption is beneficial (DICE, FMA, KPOLY5, KT4P13, KPHP5), we see that the MORE strategy always achieves the highest

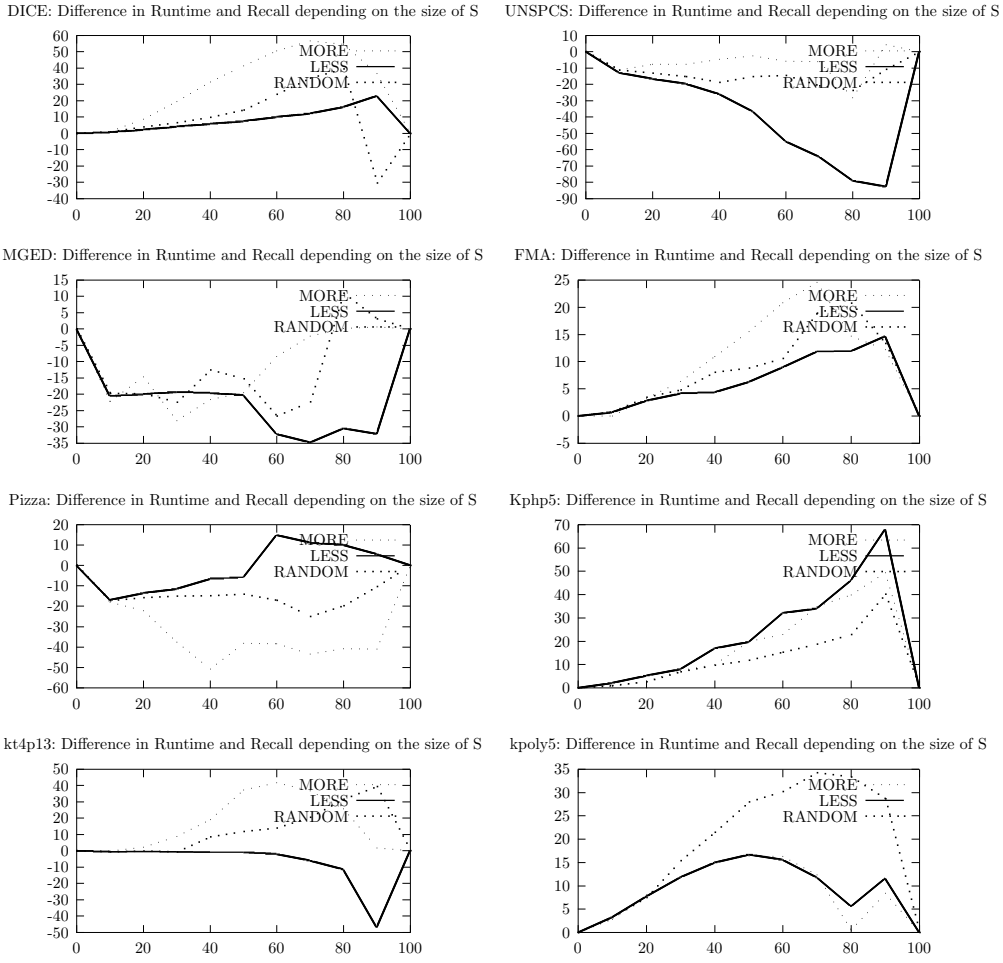

Fig. 4. Comparison of strategies

gains during almost all iterations of the anytime algorithm for DICE, FMA and KT4P13, while this is not the case for KPHP5 (LESS) and KPOLY5 (RANDOM). Some clarification can be gained when taking some more information about the ontologies into account, namely the distribution of atomic concepts over the axioms.

Ontology	freq	#	freq	#	freq	#
Kpoly5	7	7	6	8	5	1
Kt4p13	104	1	11	1	5	1
Kphp5	6	30	2	240	1	1
DICE	511	1	461	1	272	1
MGED	35	1	34	1	27	1
UNSPCS	53	1	49	1	45	1
FMA	756	1	400	1	394	1
PIZZA	72	1	54	1	53	5

Table 2. Distribution of atomic concepts over the axioms

Table 2 summarises the frequency of the most occurring atomic concepts. For example, for DICE, the single most frequent concept name occurs 511 times, whereas for Kphp5 there are 30 atoms occurring 30 times each. Only the three highest frequencies and the number of concepts with those frequencies are shown.

Now, a clear pattern can be recognised: for those ontologies, for which there are atoms occurring significantly more often then others, the MORE strategy has the best performance (DICE, FMA, KT4P13, UNSPCS), For the other ontologies the results are less conclusive: though one would guess that the equal distribution for KPOLY5 lends itself for a RANDOM approach. Little can be said about the reason why for PIZZA and Kphp5 the LESS strategy is best, apparently, there must be a rare concept with high influence on the classification.

The conclusion seems to be that the MORE strategy works best when the frequency table for concepts is skewed (i.e. an uneven distribution of the frequency of occurrence of concepts in the axioms).

4 Conclusion

In this paper we have presented a method for approximate subsumption based on ignoring a selected part of the set of concepts in an ontology. By incrementally increasing this set, we can construct an anytime algorithm for approximate subsumption, which yields sound but possibly incomplete answers w.r.t. classical subsumption queries. The behaviour of this approximate algorithm is dependent on the strategy for selecting the subset of concepts taken into account by the approximation.

Based on the formal definitions we have shown how this approximate subsumption on an ontology can be computed in terms of a classical subsumption check on a rewritten version of the ontology.

In the second half of the paper, we have applied this approximate subsumption method to a set of 8 realistich benchmark ontologies, comparing its performance against classical subsumption in terms of runtime and recall (since the approximation is sound, precision is always guaranteed). For each of these 8 benchmark ontologies we have applied three different strategies for selecting the approximated vocabulary, based on the occurrence frequency of the concepts in the axioms (most frequent first, least frequent first, random).

Our experiments show that the gain in runtime outweights the loss in recall on 5 out of 8 cases. Furthermore, the gain is larger for ontologies where classical subsumption queries are expensive. Hence, our approximation algorithm works best when it is most needed, namely on complex ontologies. Our experiments also show that of the three strategies we considered, the most-frequent-first strategy performed best, in partical on those ontologies with a skewed occurrence frequency of the concepts in the axions.

Important issues for future work are to extend our approximation to deal with OWL ontologies (instead of being restricted to $\mathcal{ALC}$) and the study of other heuristics for influencing the behaviour of the approximation, in particular heuristics that take into account the structure and complexity of the ontology.

References

1. Schaerf, M., Cadoli, M.: Tractable reasoning via approximation. Artificial Intelligence **74** (1995) 249–310
2. Stuckenschmidt, H.: Partial matchmaking using approximate subsumption. In: Proceedings of the Twenty-Second Conference on Artificial Intelligence (AAAI-07). (2007)
3. Groot, P., Stuckenschmidt, H., Wache, H.: Approximating description logic classification for semantic web reasoning. In Gómez-Pérez, A., Euzenat, J., eds.: ESWC. Volume 3532 of Lecture Notes in Computer Science., Springer (2005) 318–332
4. Baader, F., Calvanese, D., McGuinness, D.L., Nardi, D., Patel-Schneider, P.F., eds.: The Description Logic Handbook: Theory, Implementation, and Applications. Cambridge University Press (2003)