MORe: a Modular OWL Reasoner for Ontology Classification

Ana Armas Romero, Bernardo Cuenca Grau,
Ian Horrocks, Ernesto Jiménez-Ruiz

Department of Computer Science, University of Oxford

Abstract. MORe exploits module extraction techniques to divide the workload of ontology classification between two reasoners: a reasoner for the lightweight profile EL of OWL 2, and a fully fledged OWL 2 reasoner. This division is carried out in such a way that the bulk of the workload is assigned, as much as possible, to the OWL 2 EL reasoner, in order to exploit the more efficient classification techniques specific to this profile.

1 Introduction

MORe [1] is an OWL 2 reasoning system dedicated to ontology classification that integrates a general purpose OWL 2 reasoner (OWL reasoner for short) and a reasoner specific for the OWL 2 EL[1] profile (EL reasoner for short). The current implementation of MORe uses ELK [8] as its EL reasoner, and offers the possibility to choose between two OWL reasoners: HermiT 1.3.7 [4] and Pellet 2.3.0 [12]. The EL and OWL reasoners are, however, integrated in a "black-box" way: our implementation of MORe provides the required infrastructure to bundle any other OWL and/or EL reasoner.

Given an input ontology $\mathcal{O}$, MORe identifies a part of the classification of $\mathcal{O}$ that can be computed using the EL reasoner and limits the use of the OWL reasoner to a fragment of $\mathcal{O}$ as restricted as possible. The main advantage of MORe lies in its "pay-as-you-go" behaviour when an OWL 2 EL ontology is extended with axioms in a more expressive logic: the use of an efficient EL reasoner is not necessarily precluded by the extension; in fact, it is to be expected that the EL reasoner will still perform most of the computational work.

MORe performs only terminological reasoning and ignores any assertional axioms that the input ontology might contain. Therefore, completeness is only guaranteed for ontologies that contain no ABox assertions.

MORe[2] is implemented in Java using the OWL API[3] [6]. It can therefore process ontologies in any format handled by the OWL API, such as RDF/XML, OWL Functional Syntax, or OBO. It is available both as a Java library and a Protégé[4] plugin, and it can also be used via a commmand line interface.

[1] http://www.w3.org/TR/owl2-profiles/#OWL_2_EL
[2] https://code.google.com/p/more-reasoner/
[3] http://owlapi.sourceforge.net/
[4] http://protege.stanford.edu/

2 The Technique

The main idea behind the technique implemented in MORe is to identify, given an ontology $\mathcal{O}$ with signature $\mathsf{Sig}(\mathcal{O})$, two subsets $\mathcal{M}_1$, $\mathcal{M}_2$ of $\mathcal{O}$ such that

- $\mathcal{M}_1$ is as small as possible;
- the output of classifying $\mathcal{M}_2$ with the EL reasoner is *complete* for $\mathsf{Sig}(\mathcal{M}_2)$ w.r.t. $\mathcal{O}$ (i.e. it contains all subsumption relations $\mathsf{A} \sqsubseteq \mathsf{B}$ entailed by $\mathcal{O}$ such that $\mathsf{A} \in \mathsf{Sig}(\mathcal{M}_2)$);
- the output of classifying $\mathcal{M}_1$ with the OWL reasoner is *complete* for $\mathsf{Sig}(\mathcal{M}_1)$ w.r.t. $\mathcal{O}$; and
- $\mathsf{Sig}(\mathcal{M}_1) \cup \mathsf{Sig}(\mathcal{M}_2) = \mathsf{Sig}(\mathcal{O})$.

Our implementation of MORe relies on ELK, which does not yet implement the whole of OWL 2 EL. The unsupported constructs are documented and hence we can identify the fragment $\mathcal{L}_{\text{ELK}}$ of OWL 2 EL implemented by ELK.

They key to identifying $\mathcal{M}_1$ and $\mathcal{M}_2$ is in computing an $\mathcal{L}_{ELK}$-*signature*: a signature $\Sigma^{\text{ELK}} \subseteq \mathsf{Sig}(\mathcal{O})$ such that the $\bot$-*module* for Σ^{ELK} in $\mathcal{O}$ is an ontology in the language $\mathcal{L}_{\text{ELK}}$ for which ELK is complete.

The $\bot$-module for $\mathcal{O}$ and Σ, $\mathcal{M}_{[\mathcal{O},\Sigma]}$, is the smallest subset of $\mathcal{O}$ such that all axioms in $\mathcal{O} \setminus \mathcal{M}_{[\mathcal{O},\Sigma]}$ are $\bot$-local w.r.t. $\Sigma \cup \mathsf{Sig}(\mathcal{M}_{[\mathcal{O},\Sigma]})$. Intuitively, an axiom α is $\bot$-local w.r.t. Σ if replacing by $\bot$ all occurrences in α of symbols not in Σ would turn α into a syntactically recognisable tautology; e.g., the axiom $\mathsf{A} \sqsubseteq \mathsf{B}$ is $\bot$-local w.r.t. $\Sigma = \{\mathsf{B}\}$. Cuenca Grau et al. [2] offer a deeper insight into the notions of $\bot$-module, $\bot$-locality, and modularity in a more general sense. For the scope of this system description, we only remark the following properties:

1. For any class A in $\mathsf{Sig}(\mathcal{M}_{[\mathcal{O},\Sigma]})$:
   (a) if A is unsatisfiable in $\mathcal{O}$ then it is also unsatisfiable in $\mathcal{M}_{[\mathcal{O},\Sigma]}$
   (b) if another class B in $\mathsf{Sig}(\mathcal{O})$ is a superclass of A in $\mathcal{O}$, then it is so in $\mathcal{M}_{[\mathcal{O},\Sigma]}$ as well —and so B is in $\mathsf{Sig}(\mathcal{M}_{[\mathcal{O},\Sigma]})$ too.
2. If $\Sigma_1 \subseteq \Sigma_2$, then if some axiom α is $\bot$-local w.r.t. Σ_2, it is also $\bot$-local w.r.t. Σ_1, and therefore $\mathcal{M}_{[\mathcal{O},\Sigma_1]} \subseteq \mathcal{M}_{[\mathcal{O},\Sigma_2]}$.
3. both checking $\bot$-locality and extracting a $\bot$-module can be done in polynomial time.

Property 1(b), in particular, is not shared by other kinds of modules, and makes $\bot$-modules especially well suited for classification purposes.

2.1 Modular Combination of Reasoners

The integration of the two reasoners is performed as follows. Given an OWL 2 ontology $\mathcal{O}$, MORe first tries to compute a nonempty $\mathcal{L}_{\text{ELK}}$-signature Σ^{ELK} for $\mathcal{O}$ (details of how this is done are given in Section 2.2). If it suceeds, then ELK is used to classify $\mathcal{M}_{[\mathcal{O},\Sigma^{\text{ELK}}]}$, and HermiT or Pellet to classify $\mathcal{M}_{[\mathcal{O},\mathsf{Sig}(\mathcal{O})\setminus\Sigma^{\text{ELK}}]}$; finally, both partial hierarchies are unified into a single one. If MORe fails to find a nonempty $\mathcal{L}_{\text{ELK}}$-signature, then it delegates the whole classification to either HermiT or Pellet. Details about the correctness (soundness and completeness) of this technique can be found in Armas Romero et al. [1].

2.2 Computing an $\mathcal{L}_{\text{ELK}}$-signature

To find a suitable $\mathcal{L}_{\text{ELK}}$-signature Σ^{ELK} for a given ontology $\mathcal{O}$, MORe first identifies the set $\mathcal{S}$ of axioms that ELK cannot process, and —if possible— a subset Σ of $\mathsf{Sig}(\mathcal{O})$ such that all the axioms in $\mathcal{S}$ are $\bot$-local w.r.t. Σ. This alone, however, does not guarantee that $\mathcal{M}_{[\mathcal{O},\Sigma]} \cap \mathcal{S} = \emptyset$.

Example 1. Consider the ontology $\mathcal{O}_{\mathsf{ex}}$ consisting of the following axioms:

$$\mathsf{A} \equiv \mathsf{B} \sqcup \mathsf{C} \qquad \mathsf{B} \equiv \mathsf{D} \sqcap \exists \mathsf{R}.\mathsf{E} \qquad \mathsf{F} \sqsubseteq \exists \mathsf{R}.\mathsf{G}$$

All the axioms in $\mathcal{O}_{\mathsf{ex}}$ are in $\mathcal{L}_{\text{ELK}}$ except for $\alpha = \mathsf{A} \equiv \mathsf{B} \sqcup \mathsf{C}$. Now, we have that α is $\bot$-local w.r.t. a signature Σ iff $\Sigma \cap \{\mathsf{A}, \mathsf{B}, \mathsf{C}\} = \emptyset$, therefore, α is $\bot$-local w.r.t. $\Sigma = \mathsf{Sig}(\mathcal{O}_{\mathsf{ex}}) \setminus \{\mathsf{A}, \mathsf{B}, \mathsf{C}\}$. However, $\beta = \mathsf{B} \equiv \mathsf{D} \sqcap \exists \mathsf{R}.\mathsf{E}$ is not $\bot$-local w.r.t. Σ, so $\beta \in \mathcal{M}_{[\mathcal{O},\Sigma]}$ and $\mathsf{B} \in \mathsf{Sig}(\mathcal{M}_{[\mathcal{O},\Sigma]})$, and therefore α is not $\bot$-local w.r.t. $\Sigma \cup \mathsf{Sig}(\mathcal{M}_{[\mathcal{O},\Sigma]})$ and needs to be in $\mathcal{M}_{[\mathcal{O},\Sigma]}$. ♢

All we need to do is progressively reduce Σ until $\mathsf{Sig}(\mathcal{M}_{[\mathcal{O},\Sigma]}) \subseteq \Sigma$. This can be done as follows:

1. Let $\mathcal{S}_0$ be the set of axioms in $\mathcal{O}$ that are not in $\mathcal{L}_{\text{ELK}}$ and let $\Sigma_0 = \mathsf{Sig}(\mathcal{O})$.
2. Reduce Σ_0 to some $\Sigma_1 \subset \Sigma_0$ such that $\mathcal{S}_0$ is $\bot$-local w.r.t. Σ_1. If this is not possible, then make $\Sigma_1 = \emptyset$.
3. Compute the set $\mathcal{S}_1$ of axioms in $\mathcal{M}_{[\mathcal{O},\Sigma_1]}$ containing symbols outside Σ_1.
4. Repeat Steps 2–3 until $\mathcal{S}_i = \emptyset$ (i.e. until $\mathsf{Sig}(\mathcal{M}_{[\mathcal{O},\Sigma_i]}) \subseteq \Sigma_i$) or $\Sigma_i = \emptyset$.

It is important to note that, in some cases, there may be several different ways of obtaining Σ_{i+1} from Σ_i.

Example 2. As shown in the previous example, taking $\Sigma = \mathsf{Sig}(\mathcal{O}_{\mathsf{ex}}) \setminus \{\mathsf{A}, \mathsf{B}, \mathsf{C}\}$ is not enough to keep $\mathsf{A} \equiv \mathsf{B} \sqcup \mathsf{C}$ outside $\mathcal{M}_{[\mathcal{O},\Sigma]}$. We need to remove more symbols from Σ to keep $\mathsf{B} \equiv \mathsf{D} \sqcap \exists \mathsf{R}.\mathsf{E}$ outside $\mathcal{M}_{[\mathcal{O},\Sigma]}$ too. One possibility would be to remove D from Σ, but we could also choose to remove R or E instead. It turns out that choosing one option over another can change things substantially.

If we chose to take $\Sigma_1 = \mathsf{Sig}(\mathcal{O}_{\mathsf{ex}}) \setminus \{\mathsf{A}, \mathsf{B}, \mathsf{C}, \mathsf{R}\}$ then, because $\mathsf{F} \sqsubseteq \exists \mathsf{R}.\mathsf{G}$ is not $\bot$-local w.r.t. Σ_1 and contains the symbol R, we would need to further reduce Σ_1 to some $\Sigma_2 \subset \Sigma_1$.

However, if we took $\Sigma_1 = \mathsf{Sig}(\mathcal{O}_{\mathsf{ex}}) \setminus \{\mathsf{A}, \mathsf{B}, \mathsf{C}, \mathsf{D}\}$ or $\Sigma_1 = \mathsf{Sig}(\mathcal{O}_{\mathsf{ex}}) \setminus \{\mathsf{A}, \mathsf{B}, \mathsf{C}, \mathsf{E}\}$, then we would already have $\mathsf{Sig}(\mathcal{M}_{[\mathcal{O}_{\mathsf{ex}},\Sigma_1]}) = \Sigma_1$, and we would be done. ♢

The $\bot$-module $\mathcal{M}_{[\mathcal{O},\mathsf{Sig}(\mathcal{O}) \setminus \Sigma^{\text{ELK}}]}$ that the OWL reasoner needs to classify is likely to be smaller the larger Σ^{ELK} is. Therefore, it is desirable to find heuristics to choose each Σ_i in a way that leads to an $\mathcal{L}_{\text{ELK}}$-signature as large as possible. Below we describe the main heuristics that we have implemented in MORe.

Keeping Properties As far as possible, we try not to remove properties from Σ_i. The reason for this is that most ontologies contain fewer properties than classes, and each property usually appears in more axioms than any class. Thus, removing a property from Σ_i is more likely to bring more axioms into the next $\mathcal{S}_{i+1}$ and lead to a smaller $\mathcal{L}_{\text{ELK}}$-signature.

Global Symbols We perform a preprocessing stage to identify a (possibly empty) set Γ of *global symbols* in $\mathsf{Sig}(\mathcal{O})$, such that either $\Gamma \subseteq \Sigma^{\text{ELK}}$ or $\Sigma^{\text{ELK}} = \emptyset$. For this, we first find the set $\mathcal{G}$ of all *global axioms* in $\mathcal{O}$, i.e. those that cannot possibly be made $\bot$-local, (e.g. axioms of the form $\top \sqsubseteq \mathsf{C}$) and take $\Gamma = \mathsf{Sig}(\mathcal{G})$. We then keep adding to Γ the signatures of all the axioms in $\mathcal{O}$ that would only be $\bot$-local w.r.t. Σ^{ELK} if some symbol in Γ was left outside Σ^{ELK}, until no more symbols need to be added to Γ.

Then, we will only ever consider sets Σ_i such that $\Gamma \subseteq \Sigma_i$. As the following example shows, this can sometimes mark the difference between finding a non-empty $\mathcal{L}_{\text{ELK}}$-signature or not.

Example 3. Consider the ontology $\mathcal{O}'_{\mathsf{ex}} = \mathcal{O}_{\mathsf{ex}} \cup \{\top \sqsubseteq \exists \mathsf{R}.\mathsf{E}\}$. In the previous example we saw how both $\mathsf{Sig}(\mathcal{O}_{\mathsf{ex}}) \setminus \{\mathsf{A}, \mathsf{B}, \mathsf{C}, \mathsf{D}\}$ and $\mathsf{Sig}(\mathcal{O}_{\mathsf{ex}}) \setminus \{\mathsf{A}, \mathsf{B}, \mathsf{C}, \mathsf{E}\}$ were equally good choices when choosing a suitable Σ_1 for $\mathcal{O}_{\mathsf{ex}}$. This is not the case any more with $\mathcal{O}'_{\mathsf{ex}}$, as after choosing $\mathsf{Sig}(\mathcal{O}'_{\mathsf{ex}}) \setminus \{\mathsf{A}, \mathsf{B}, \mathsf{C}, \mathsf{E}\}$ we would need to try to keep $\top \sqsubseteq \exists \mathsf{R}.\mathsf{E}$ outside $\mathcal{M}_{[\mathcal{O}'_{\mathsf{ex}}, \Sigma^{\text{ELK}}]}$ too; but this is not possible, so in the end we would have $\Sigma^{\text{ELK}} = \emptyset$. ♢

Reducing Nondeterminism In each iteration of the algorithm, instead of considering the set $\mathcal{S}_i$ as a whole, we split it into two subsets: $\mathcal{S}_i^{nondet}$, containing those axioms in $\mathcal{S}$ for which there are several ways in which Σ_i can be reduced to make them $\bot$-local, and $\mathcal{S}_i^{det}$, those for which there is only one way.

Whenever $\mathcal{S}_i^{det} \neq \emptyset$, we obtain Σ_{i+1} by removing from Σ_i the symbols required by each axiom in $\mathcal{S}_i^{det}$, and ignore $\mathcal{S}_i^{nondet}$. When $\mathcal{S}_i^{det} = \emptyset$, we deal with the axioms in $\mathcal{S}_i^{nondet}$ taking a greedy approach —finding the optimal solution is often too expensive.

The intuition behind this heuristic is that, by postponing making any nondeterministic decisions as much as possible, we might eliminate the need to make them altogether.

Note that, using this heuristic, we are not guaranteed to handle all the non $\mathcal{L}_{\text{ELK}}$-axioms in the first iteration any more, therefore we also have to consider in each $\mathcal{S}_i$ those non-$\mathcal{L}_{\text{ELK}}$ axioms that are still not $\bot$-local w.r.t. Σ_i.

Example 4. Consider the ontology $\mathcal{O}''_{\mathsf{ex}}$ consisting of all the axioms in $\mathcal{O}_{\mathsf{ex}}$, plus the following additional axioms:

$$\mathsf{E} \sqsubseteq \mathsf{C} \qquad \mathsf{H} \sqsubseteq \exists \mathsf{R}.\mathsf{E} \qquad \mathsf{I} \equiv (\mathsf{E} \sqcap \mathsf{F}) \sqcup (\mathsf{G} \sqcap \mathsf{H})$$

We first get $\mathcal{S}_0^{nondet} = \{\mathsf{I} \equiv (\mathsf{E} \sqcap \mathsf{F}) \sqcup (\mathsf{G} \sqcap \mathsf{H})\}$ and $\mathcal{S}_0^{det} = \{\mathsf{A} \equiv \mathsf{B} \sqcup \mathsf{C}\}$. The new non-$\mathcal{L}_{\text{ELK}}$ axiom, $\mathsf{I} \equiv (\mathsf{E} \sqcap \mathsf{F}) \sqcup (\mathsf{G} \sqcap \mathsf{H})$, goes into $\mathcal{S}_0^{nondet}$ because it could be handled by removing any of the following sets of symbols: $\{\mathsf{I}, \mathsf{E}, \mathsf{G}\}$, $\{\mathsf{I}, \mathsf{F}, \mathsf{G}\}$, $\{\mathsf{I}, \mathsf{E}, \mathsf{H}\}$ or $\{\mathsf{I}, \mathsf{F}, \mathsf{H}\}$. For now we only deal with $\mathcal{S}_0^{det} = \{\mathsf{A} \equiv \mathsf{B} \sqcup \mathsf{C}\}$, and we do so by taking $\Sigma_0 = \mathsf{Sig}(\mathcal{O}''_{\mathsf{ex}}) \setminus \{\mathsf{A}, \mathsf{B}, \mathsf{C}\}$.

Then we obtain the sets $\mathcal{S}_1^{nondet} = \{\mathsf{B} \equiv \mathsf{D} \sqcap \exists \mathsf{R}.\mathsf{E}, \mathsf{I} \equiv (\mathsf{E} \sqcap \mathsf{F}) \sqcup (\mathsf{G} \sqcap \mathsf{H})\}$ and $\mathcal{S}_1^{det} = \{\mathsf{E} \sqsubseteq \mathsf{C}\}$ and we deal with $\mathsf{E} \sqsubseteq \mathsf{C}$ by taking $\Sigma_1 = \mathsf{Sig}(\mathcal{O}''_{\mathsf{ex}}) \setminus \{\mathsf{A}, \mathsf{B}, \mathsf{C}, \mathsf{E}\}$.

Table 1. Ontology metrics

Ontology \ Metrics	Expressivity	$\lvert\mathsf{Sig}(\mathcal{O})\rvert$	$\lvert\mathcal{O}\rvert$	$\lvert\mathcal{O} \setminus \mathcal{O}_{\mathcal{L}_{\text{ELK}}}\rvert$	$\lvert\mathcal{M}_{\text{OWL2}}\rvert$
Gazetteer	$\mathcal{ALE}+$	517,039	652,361	0	0%
Cardiac Electrophys.	$\mathcal{SHF}(\mathcal{D})$	81,020	124,248	22	1%
Protein	$\mathcal{S}$	35,205	46,114	15	22%
Biomodels	$\mathcal{SRIF}$	187,577	439,248	22,104	45%
Cell Cycle v0.95	$\mathcal{SRI}$	144,619	511,354	1	<0.1%
Cell Cycle v2.01	$\mathcal{SRI}$	106,517	624,702	9	98%
NCI v09.12d	$\mathcal{SH}(\mathcal{D})$	77,571	109,013	4,682	58%
NCI v13.03d	$\mathcal{SH}(\mathcal{D})$	97,652	136,902	158	57%
$\text{SNOMED}_{15\sqcup}$	$\mathcal{ALCR}$	291,216	291,185	15	3%
SNOMED+LUCADA	$\mathcal{ALCRIQ}(\mathcal{D})$	309,405	550,453	122	0.1%

In the next iteration, we get $\mathcal{S}_2^{nondet} = \{\mathsf{I} \equiv (\mathsf{E} \sqcap \mathsf{F}) \sqcup (\mathsf{G} \sqcap \mathsf{H})\}$ —note that axiom $\mathsf{B} \equiv \mathsf{D} \sqcap \exists \mathsf{R}.\mathsf{E}$ has been taken care of indirectly— and $\mathcal{S}_2^{det} = \{\mathsf{H} \sqsubseteq \exists \mathsf{R}.\mathsf{E}\}$, and we handle $\mathsf{H} \sqsubseteq \exists \mathsf{R}.\mathsf{E}$ by taking $\Sigma_2 = \mathsf{Sig}(\mathcal{O}''_{\mathsf{ex}}) \setminus \{\mathsf{A}, \mathsf{B}, \mathsf{C}, \mathsf{E}, \mathsf{H}\}$.

After that, we find $\mathcal{S}_2^{nondet} = \emptyset$ and $\mathcal{S}_2^{det} = \{\mathsf{I} \equiv (\mathsf{E} \sqcap \mathsf{F}) \sqcup (\mathsf{G} \sqcap \mathsf{H})\}$, and take $\Sigma_3 = \mathsf{Sig}(\mathcal{O}''_{\mathsf{ex}}) \setminus \{\mathsf{A}, \mathsf{B}, \mathsf{C}, \mathsf{E}, \mathsf{H}, \mathsf{I}\}$, which finally gives $\mathcal{S}_3 = \emptyset$. Thus, we have computed $\Sigma^{\text{ELK}} = \Sigma_3$ without making any nondeterministic decisions.

3 Evaluation

We have tested MORe using an Ubuntu 12.04 64-bit machine with 7.8 GiB of RAM (fully assigned to the JVM) and an Intel Core i7-3770 CPU @ 3.40GHz x 8 processor. Our test ontology suite includes six BioPortal ontologies[5] [3] —for Biomodels we consider only its TBox—, two different versions of NCI[6] [5], and two extensions of SNOMED[7] [9]: $\text{SNOMED}_{15\sqcup}$ was built from a 2012 version of SNOMED, following the suggestions of domain experts, by adding 15 axioms containing disjunctions; SNOMED+LUCADA was obtained by mapping a 2011 version of SNOMED to the terminological part of the LUCADA ontology[8] [10, 11] using the ontology matching system LogMap [7]. Table 1 gives an overview of the general features of these ontologies, including the number of non-$\mathcal{L}_{\text{ELK}}$ axioms they contain and the size of the module extacted by MORe for the OWL reasoner, $\mathcal{M}_{[\mathcal{O},\mathsf{Sig}(\mathcal{O})\setminus\Sigma^{\text{ELK}}]}$, referred to as $\mathcal{M}_{\text{OWL2}}$ in Table 1.

[5] http://bioportal.bioontology.org/ontologies/1397
[6] http://evs.nci.nih.gov/ftp1/NCI_Thesaurus/archive
[7] http://www.ihtsdo.org/snomed-ct/
[8] The LUCADA ontology ($\mathcal{ALCHI}(\mathcal{D})$) contains 476 entities and can be classified by both HermiT and Pellet in less than 2 seconds

Table 2. Classification times in seconds

Ontology \ Reasoner	MORe$_{\text{HermiT}}$		HermiT	MORe$_{\text{Pellet}}$		Pellet
	HermiT	total		Pellet	total	
Gazetteer	0	20.6	651	0	20.3	1,414
Cardiac Electrophys.	0.3	6.3	22.7	0.3	5.5	11.0
Protein	2.0	4.8	10.0	2.0	4.7	2,920
Biomodels	377	487	582	373	483	1,915
Cell Cycle v0.95	<0.1	9.9	mem	<0.1	10.4	3,433
Cell Cycle v2.01	mem	mem	mem	mem	mem	3,435
NCI v09.12d	244	252	261	256	266	93.6
NCI v13.03d	45.1	62.7	68.4	45.7	62.9	191
SNOMED$_{15\sqcup}$	4.5	25.4	1,395	4.4	22.9	4,314
SNOMED+LUCADA	1.1	28.8	1,302	1.2	29.2	mem

We analyse our results by comparing the performance of MORe using HermiT vs. HermiT alone, and of MORe using Pellet vs. Pellet alone. A summary of all results can be found in Table 2. mem indicates an out of memory error.

When integrating HermiT, MORe is always able to improve, or at least maintain its performance. We remark the case of Cell Cycle v0.95, where the performance is improved from an out of memory error to termination in under 10s.

Integrating Pellet, however, sometimes has an unexpected effect. In the cases of NCI v09.12d and Cell Cycle v2.01, Pellet takes longer to classify $\mathcal{M}_{\text{OWL2}}$ when integrated in MORe than to classify the whole ontology on its own. This however, does not happen when Pellet is used to classify $\mathcal{M}_{\text{OWL2}}$ independently of MORe. We are still unsure about the causes of this phenomenon. Apart from these two cases, MORe is still often able to improve on the performance of Pellet.

It is worth mentioning that the reason why, in the case of Cell Cycle v2.01, the portion of the ontology that the OWL reasoner has to process is so close to the whole ontology (98%) is because the 9 non $\mathcal{L}_{\text{ELK}}$ axioms are symmetric property axioms, which force their 9 respective properties out of Σ^{ELK}, reducing it to a very small set.

4 Conclusions and Future Directions

We are continuing to develop MORe, exploring new ways of further reducing the workload assigned to a general purpose OWL 2 classification algorithm. We are looking into the possibility of alternative modularity notions specific for this application, and also into exploiting computational properties of other lightweight ontology languages to combine our modular approach with one based on finding lower and upper bounds for the classification.

Acknowledgements

This work was supported by the Royal Society, the Seventh Framework Program (FP7) of the European Commission under Grant Agreement 318338, "Optique", and the EPSRC projects Score!, ExODA and MaSI^3.

References

1. Armas Romero, A., Cuenca Grau, B., Horrocks, I.: MORe: Modular Combination of OWL Reasoners for Ontology Classification. In: International Semantic Web Conference (ISWC). pp. 1–16 (2012)
2. Cuenca Grau, B., Horrocks, I., Kazakov, Y., Sattler, U.: Modular reuse of ontologies: Theory and practice. J. Artificial Intelligence Research 31, 273–318 (2008)
3. Fridman Noy, N., Shah, N.H., Whetzel, P.L., Dai, B., Dorf, M., Griffith, N., Jonquet, C., Rubin, D.L., Storey, M.A.D., Chute, C.G., Musen, M.A.: BioPortal: ontologies and integrated data resources at the click of a mouse. Nucleic Acids Research 37(Web-Server-Issue), 170–173 (2009)
4. Glimm, B., Horrocks, I., Motik, B., Shearer, R., Stoilos, G.: A novel approach to ontology classification. J. of Web Semantics 10(1) (2011)
5. Golbeck, J., Fragoso, G., Hartel, F.W., Hendler, J.A., Oberthaler, J., Parsia, B.: The National Cancer Institute's Thésaurus and Ontology. J. Web Semantics 1(1), 75–80 (2003)
6. Horridge, M., Bechhofer, S.: The OWL API: A Java API for OWL ontologies. Semantic Web 2(1), 11–21 (2011)
7. Jiménez-Ruiz, E., Cuenca Grau, B., Zhou, Y., Horrocks, I.: Large-scale interactive ontology matching: Algorithms and implementation. In: European Conference on Artificial Intelligence (ECAI). pp. 444–449 (2012)
8. Kazakov, Y., Krötzsch, M., Simancik, F.: Concurrent classification of EL ontologies. In: International Semantic Web Conference (ISWC). pp. 305–320 (2011)
9. Schulz, S., Cornet, R., Spackman, K.A.: Consolidating SNOMED CT's ontological commitment. Applied Ontology 6(1), 1–11 (2011)
10. Sesen, M.B., Bañares-Alcántara, R., Fox, J., Kadir, T., Brady, J.M.: Lung Cancer Assistant: An Ontology-Driven, Online Decision Support Prototype for Lung Cancer Treatment Selection. In: OWL: Experiences and Directions Workshop (2012)
11. Sesen, M.B., Jiménez-Ruiz, E., Bañares-Alcántara, R., Brady, J.M.: Evaluating OWL 2 Reasoners in the context of Clinical Decision Support in Lung Cancer Treatment Selection. In: OWL Reasoner Evaluation (ORE) Workshop (2013)
12. Sirin, E., Parsia, B., Cuenca Grau, B., Kalyanpur, A., Katz, Y.: Pellet: A practical OWL DL reasoner. J. of Web Semantics 5(2), 51–53 (2007)