Efficient Projection of Ontologies

Julius Köpke, Johann Eder and Michaela Schicho

Author's Version of

Julius Köpke, Johann Eder, and Michaela Schicho. “Efficient Projection of Ontologies”.
In: On the Move to Meaningful Internet Systems: OTM 2013 Conferences. Ed. by Robert Meersman, Hervé Panetto, Tharam Dillon, Johann Eder, Zohra Bellahsene, Norbert Ritter, Pieter De Leenheer, and Deijing Dou. Vol. 8185. Lecture Notes in Computer Science. Heidelberg: Springer, 2013, pp. 659–676.
URL: http://dx.doi.org/10.1007/978-3-642-41030-7_48.

Efficient Projection of Ontologies

Julius Köpke, Johann Eder and Michaela Schicho

Department of Informatics-Systems, Alpen-Adria Universität Klagenfurt, Austria
firstname.lastname@aau.at http://www.aau.at/isys

Abstract. In collaborative and interoperable information systems it is frequently necessary to export a subset of an ontology, in particular to export all instances and axioms subsumed by a shared domain ontology. Technically this can be realized by projecting the Abox axioms of the private instances to the vocabulary of the shared ontology using a reasoner. We show that the direct application of this method results in a very redundant export. Therefore, we propose methods to generate a much smaller equivalent export in an efficient way. These methods are based on a deep analysis of properties of axiom sets that can be used to efficiently filter redundancies during the export process and provide methods to further minimize redundancies in the result. We evaluate the performance and the degree of minimality that can be reached with our approach with standard benchmarking ontologies with large Aboxes and a use case of the bio-medical domain.

Key words: OWL, Ontology, Abox export, Abox minimizing, instance export, instance migration

1 Introduction

For cooperative information systems it is frequently necessary to export parts of an ontology from one information system and import it in another (e.g. shared) information system. Here we focus on a particular situation where a local (or private) ontology is an extension of a shared (domain) ontology. The reasons for this situation may be manifold: the local information system might include other (upper) domain ontologies as well, the local ontology might be more fine grained than the shared ontology, the information shared should not be too detailed e.g. to protect trade secrets or privacy.

We motivate the projection of ontologies with a concrete use case: the sharing of data in federated biobanks. Biobanks are collections of biological material (Tissue, body fluids, cell cultures, etc.) and data describing this material and their donors (e.g health records, lab data, -omics data) [2, 26]. Biobanks are very valuables resources for life science research providing material and data for biomedical research projects and studies. Currently, great efforts are undertaken to link biobanks creating an interconnected federated resource for biomedical research [25, 7].

Creating a network of federated biobanks [21, 7] requires the integration of heterogenous, autonomous, individually developed biobank databases [9, 10]. Experiments based on data of (unknowingly) poor quality yield incorrect results [24]. One of the greatest challenges is to assure that data is correctly interpreted, data is correctly transformed between the different sites and it is possible to decide whether data stemming from different sites is comparable and can be used together for performing a biomedical study. This can be achieved by sharing not only data, but also the provenance [22] of the data and of the material in a provenance ontology. The provenance of a data item is the process which lead to this data item [4]. Thus, provenance is based on the question: *Who* has done something; *how, where*, and *when* was it done and based on *what*? As a consequence, provenance information is conceptualized using five interconnected elements including where, who, how, when and what [19] and is frequently represented in form of an ontology. However, since the local provenance ontology might contain more fine grained information, and not all information might be allowed to be shared for privacy reasons and ethical, and legal concerns [8], it is necessary to project the provenance information to the shared upper ontology of the federation.

For an example: the local biobank might store information that the diagnosis was made by John Smith, and we might infer that John Smith is pathologist, he has 10 years of experience, is 40 years old, etc. For the exported provenance ontology we would like to restrict the information to the fact that the diagnosis was made by a pathologist with more than 4 years of experience. This is the relevant information for the federation, as it allows to assign a credibility measure to the diagnosis. The other information is necessary for the local information system for various reasons, but it should not be passed on to the federation.

Data should only be exported if it is mapped to the federation (or domain) ontology and can be expressed in terms of this domain ontology. If we simply remove all explicit references to elements of the private ontology we will loose too much information. When the local instance data only addresses concepts and properties of the private ontology we would in fact loose all information (for example consider a medical case that only points to a local diagnosis ontology that is mapped to a global ontology. Simply striping away local assertions will result in instances without any assertions). Instead we need to rewrite the data that is only implicitly instance-data of the domain ontology to explicit instance data of the domain ontology that does not have any references to the private ontology.

In principal we need to export all Abox axioms regarding a set of instances of a private OWL [17] (DL direct semantics) ontology that only refer to the shared ontology. Since the number of such axioms is potentially infinite we restrict the exported axioms to Abox axioms that refer only to named entities. This is directly inline with our scenario and guarantees finite sets. This also implies that we only export axioms that refer to sub- or equivalent- roles / classes of the domain ontology which is sufficient for a large class of ontologies including our use-case. In order to also export axioms that refer to super-classes /-roles of

the shared ontology the shared ontology can potentially be extended with such named concepts before the export is executed.
Creating an export in a naive way results in very large and highly redundant sets of axioms. Straightforward redundancy removals on the other hand are highly inefficient. So the goal for the research reported here was to develop a method which is both efficient and which generates export sets with low redundancy. This new method is based on an in-depth analysis which properties of ontologies lead to redundancies in the export set. Exhaustive experiments with several ontologies show that this method is much more efficient that a straightforward approach and the remaining redundancy is marginal.

2 Abox Projection

2.1 Definition

We assume that the reader is familiar with the basic definitions of OWL (DL) [17] and description logics (see [14]) and only provide the essential definitions we need for defining the export problem formally here.

Definition 1. *The signature of an ontology O* is a set of named entities. This set can be further divided into the set of named classes denoted $O.sig.classes$, the set of roles $O.sig.Roles$ and the set of named individuals $O.sig.Individuals$.

Definition 2. *We restrict Abox axioms to the following axiom types*

- Class membership: $C(a)$ the named individual a is an instance of the named class C.
- Role assertions[1]: $R(a, b)$ defines that the named individual a is connected to the named individual b with the named role R. OWL allows to define object-properties as roles between individuals and data-type property roles as roles that connect individuals and typed literals.
- Equivalent individuals: $(a \approx b)$ defines that the named individuals a and b are equivalent.
- Different individuals: $(a \not\approx b)$ defines that the named individuals a and b are explicitly not equivalent. This is important due to the absence of the unique name assumption.

Definition 3. *An export set E of an ontology O* is a user defined subset of $O.sig.Individuals$

We can now formulate the problem of the projection:

Definition 4. *Given a private ontology O_k and an export set E of O_k and a public ontology O, the projection $P(E, O_k, O)$ is defined as a set of Abox axioms PA such that all Abox-axioms of O_k that refer only to the signature of $O \cup E$ are axioms of PA.*

[1] In contrast to OWL2 [17] we do currently not address the export of negative role assertions since they are not used by any of the relevant ontologies for our scenario.

This definition regards the ontologies O_k and O as logical theories and makes no distinction between axioms that are explicitly stated and axioms that are entailed.

2.2 A naive implementation

A straightforward implementation of the projection of a sub Abox to a sub ontology in OWL according to the definition is to iterate over all individuals of E and project the inferrable Abox axioms to the signature of O. Thus, only axioms that relate to classes, roles or individuals of $O \cup E$ are added to the projection. The corresponding algorithm is shown in algorithm 1. We show here only how to handle OWL object properties (= DL roles). Nevertheless, data-type properties can be handled in analogy.

Theorem 1. *The output of alg. 1 is valid according to the problem definition.*

Proof. All possible Abox axioms over named entities are addressed and it is guaranteed that only axioms that refer to classes and roles that belong to the signature of O are added and that only instances that are $\in E$ or that are $\in O$ are added. Since the algorithm checks the axioms by querying the entailments from the reasoner it makes no difference if an axiom a was explicitly stated in O_k or if it was inferred ($O_k \models a$). Therefore, also Abox-Axioms that relates to O which can only be inferred in O_k are made explicit in the projection

2.3 Minimal Projections for Data-Exchange

Unfortunately the naive solution produces a potentially very large number of exported axioms because it makes all axioms in the projection explicit. For example, given the class expressions *doctor* $\sqsubseteq$ *Actor*, *pathologist* $\sqsubseteq$ *doctor experienced_pathologist* $\sqsubseteq$ *pathologist* and an exported individual Mr_Smith with the type assertion *experienced_pathologist(Mr_Smith)* we will get the additional explicit assertions: *Actor(Mr_Smith), doctor(Mr_Smith),and pathologist(Mr_Smith)* in the projection. While in a minimal solution there should only be the axiom *experienced_pathologist(Mr_Smith).* The restriction to named entities makes the projection finite but still much too large for an export format, where the size of the exported data matters. For exporting purposes the number of axioms should be limited to those that are essentially required to preserve all entailments that are possible in the full projection. Therefore, we now define the minimality of axioms of one individual.

Definition 5. *Minimal set of Abox axioms of an individual:* The set A_i of all Abox axioms that refer to an individual $I \in O$ is minimal if there does not exist an Abox axiom $a \in A_i$ such that $O \setminus \{a\} \models a$.

Based on this definition we can define a minimal projection:

Algorithm 1. A naive implementation of the projection

```
1: procedure NaiveProjection(E, O_k, O, PA)
2:     PA = {}
3:     for all i ∈ E do
4:         for all C ∈ O.sig.Classes do
5:             if O_k ⊨ C(i) then
6:                 PA = PA ∪ {C(i)}
7:             end if
8:         end for
9:         for all R ∈ O.sig.Roles do
10:            for all j ∈ O.sig.Individuals ∪ E do
11:                if O_k ⊨ R(i, j) then
12:                    PA = PA ∪ {R(i, j)}
13:                end if
14:            end for
15:        end for
16:        for all  j ∈ O.sig.Individuals ∪ E do
17:            if i ≠ j ∧ O_k ⊨ i ≈ j then
18:                PA = PA ∪ {i ≈ j}
19:            end if
20:            if O_k ⊨ i ≉ j then
21:                PA = PA ∪ {i ≉ j}
22:            end if
23:        end for
24:    end for
25: end procedure
```

Definition 6. *A minimal projection $P_{min}(E, O_k, O)$ is defined as a subset of of the projection $P(E, O_k, O)$ such that $P_{min}(E, O_k, O) = \bigcup_{i=1}^{|E|} A_i$ where A_i denotes one minimal set of Abox axioms of the individual $i \in P(E, O_k, O)$.*

Obviously, since there can be multiple minimal sets of Abox axioms for each individual a minimal projection is not unique. In this work we are interested in finding one minimal projection that is used for data exchange.

A black-box algorithm that computes a minimal projection for a given projection P and a target ontology O can be implemented straightforward. The corresponding algorithm is shown in algorithm 2. It is inspired by an algorithm for the construction of minimal unsatisfiable sub Tboxes of [11]. It simply iterates over all individuals in the signature of P and removes each explicitly defined axiom that can still be entailed when the explicit axiom is removed.

While this black-box solution obviously achieves a minimal projection according to the definition it has the drawback of a simply unfeasible performance. Each removal of an axiom from the projection requires to restart/re-synchronize the reasoner which is unfeasible for larger ontologies (see section 4).

Algorithm 2. Minimizing a projection P

```
1: procedure MinimizeProjection(P, O)
2:     for all i ∈ P.sig.Individuals do
3:         for all a ∈ Axioms ∈ P over i do
4:             if O \ a_i ⊨ a_i. then
5:                 remove a_i from P
6:             end if
7:         end for
8:     end for
9: end procedure
```

3 Efficient Export of sub Aboxes

The output of the naive exporter is very redundant. We could also show, that a black box minimizer is unfeasible. We are now interested in strongly reducing the number of axioms in the projection without changing the entailments. We call axioms that can be removed without modifying the entailments redundant axioms. We now discuss properties of a projection that can be exploited to generate a far less redundant projection with an incremental algorithm that aims to minimize the calls to the reasoner to provide proper scalability in section 3.1. The incremental algorithm never removes already inserted axioms from the generated output. This approach allows a very fast export but some reasons for redundancies cannot be processed in an incremental way. Therefore, we will discuss reasons for such redundancies and propose a post processing algorithm that removes such redundancies from the export in section 3.2.

3.1 Improving the generation of the projection

In this section we discuss properties of a projection that can be used to already reduce the number of redundant axioms using an incremental export algorithm.

Avoiding redundancies due to the class or property hierarchy

Theorem 2. *A type assertion for an individual x, $C(x)$ in a projection $P(E, O_k, O)$ is redundant, if there exists a type assertion $D(x) \in P(E, O_k, O)$ and the target ontology O entails $D \sqsubseteq C$.*

Proof. When the result of the projection $P(E, O_k, O)$ is merged with the target ontology O, all superclasses of C are implicitly assigned as types of x. Therefore, their explicit definition is redundant.

Theorem 3. *A role assertion $R(x, y)$ for an individual x to another individual y in a projection $P(E, O_k, O)$ is redundant, if there exists another role assertion $S(x, y)$ in $P(E, O_k, O)$ and the target ontology O entails $S \sqsubseteq R$.*

Proof. When the result of the projection $P(E, O_k, O)$ is merged with the target ontology O, the axiom $R(x, y)$ is entailed by $S(x, y)$ and $S \sqsubseteq R$. Therefore, the explicit definition of $R(x, y)$ is redundant.

Theorem 4. *An equivalent- or subclass S of a class C has always equivalent or less direct or indirect descendants than C .*

Proof. Any subclass S_2 of a subclass S_1 is always also a subclass of the superclass S of S_1. Therefore, the superclass must have more child classes than any of its subclasses. Cyclic subclass relations do not change this behavior because in DL classes that are pairwise subclasses of each other are equivalent classes. Equivalent classes have the same number of descendants.

We could simply remove the previously discussed redundancies from a given projection by a minimizer that computes the transitive reduction [1] of the type or role assertions. Algorithmically this can be realized by a nested loop over the classes or roles and corresponding entailment checks. While this is still much more feasible than the black-box minimizer it still requires a high number of reasoner calls. Therefore, we exploit theorem 4 to get a more efficient exporter: We can sort the types and roles ascending by the number of subclasses/subroles. In a next step an advanced exporter can iterate over the sorted subroles/subclasses and only insert classes or roles when no subclass or role of the current class/roles has already been inserted. For role assertions we also need to ensure, that the role points to the same entity. Our improved algorithm effectively generates the transitive reduction without computing the transitive reduction in a nested loop fashion. While this improvement already strongly avoids redundancies - especially for ontologies with big T/R-boxes, there are still other reasons for redundant type assertions:

Avoiding redundancies due to role assertions

Theorem 5. *A type assertion of an individual a, $C(a)$ is redundant in the projection $P(E, O_k, O)$, if there exists a role assertion $R(a, b)$ and C is an equivalent or superclass of the domain of R.*

Proof. The domain C of a property P is defined as $\exists P.\top \sqsubseteq C$. Consequently it is redundant to explicitly state that an individual that has a property assertion $P(a, \top)$ is of the type C.

Theorem 6. *An explicit role assertion $R(a, b)$ in a projection $P(E, O_k, O)$ is redundant, if there exists an explicit role assertion $R^{-1}(b, a)$ in $P(E, O_k, O)$.*

Proof. The proof directly follows from the definition of inverse roles.

Theorem 7. *An explicit role assertion $R(a, b)$ in a projection $P(E, O_k, O)$ is redundant, if there exists an explicit role assertion $R(b, a)$ in $P(E, O_k, O)$ and R is defined to be symmetric.*

Proof. The proof directly follows from the definition of symmetry.

Theorem 8. *For every symmetric role R, there exists an inverse role R^{-1} such that $R^{-1} = R$*

Proof. The proof directly follows from the definition of symmetric roles.

Theorem 9. *An explicit role assertion of an individual a, $R(a, b)$ is redundant in the projection $P(E, O_k, O)$, if there exists an explicit role assertion $R'(a, c)$ and $O \models \{R \equiv R'\}$ and $O \models \{b \approx c\}$*

Proof. The proof is a direct consequence of the definition of equivalence of individuals.

Theorem 10. *An explicit role assertion of an individual a, $R(a, b)$ is redundant in the projection $P(E, O_k, O)$, if the role is reflexive in O and a is in the domain of R.*

Proof. The proof is a direct consequence of the definition of reflexivity in OWL-DL: A reflexive property p with the domain d is defined as $d \sqsubseteq \exists p.Self$. Consequently the explicit definition of p to any individual of the type d is redundant.

An improved exporter can use theorem 5 by only asserting types to the exported individuals that are not equivalent to the domain of any assigned property. Theorems 6 and 8 allow us to add only property assertions to the projection, where the inverse has not yet been added. According to theorem 8 this also handles redundancies that are induced due to symmetric properties. Actually it is sufficient to only handle inverse properties. Theorem 9 and 10 allow the exporter to realize the following optimization: Assertions to reflexive roles of the form $R(a, b)$, where a is the current individual and $a \approx b$ are skipped if a is in the domain of R and role assertions are only exported once for each equivalent individual.

Avoiding redundancies due to same individuals assertions

Theorem 11. *An explicit same individual assertion $a \approx c$ in a projection $P(E, O_k, O)$ is redundant if there exists an assertion $a \approx b$ and an assertion $b \approx c$ in $P(E, O_k, O) \cup O$.*

Proof. The equivalence relation is transitive. Therefore, any assertion that is not an element of the transitive reduction is redundant.

We can now use theorem 11 for an efficient exporter that avoids the described redundancies already during the export. We need a way to ensure, that we do not export redundant relations due to transitivity. The problem here is that we cannot calculate the transitive reduction of the axioms in the projection unless the projection is complete. Fortunately, there is another property that allows an efficient export:

Theorem 12. *An explicit same individual assertion $a \approx b$ in a $P(E, O_k, O)$ is redundant if there exists an assertion $b \approx a$ in the projection.*

Proof. The equivalence relation is symmetric.

Exploiting theorem 11 and theorem 12 allows us to apply the following approach: Export only equivalence axioms for an individual j, if there exists no equivalence axiom of the form $k \approx j$ in the current set of axioms of the incrementally generated projection, where k is already an element of the projection. According to theorem 12 it is equivalent to query for $k \approx j$ or to query for $j \approx k$. Fortunately $k \approx j$ is already part of the projection because when k was exported all equivalence axioms that can be entailed in $O_k \cup E$ were added to the projection.
We can actually broaden theorem 11 to any axiom over an equivalent individual:

Theorem 13. *Given the individuals a and c, where $a \approx c$. Any explicitly stated axiom over c is redundant, if there exists a corresponding axiom over a.*

Proof. The proof is a direct consequence of the definition of equivalence.

We can now exploit theorem 13 to make the exporting algorithm even more efficient: We only export any type of axiom over an individual j, if there exists no equivalence axiom of the form $k \approx j$ in the current set of axioms of the incrementally generated projection, where k is already an element of the projection.

3.2 Efficient Removal of redundant assertions

While the discussed properties of a projection can already be used to produce a pre-minimized export, the result is not yet minimal because there are reasons for redundancies that cannot be removed during the incremental creation of the projection. For a complete minimization we can of course again use our black-box algorithm 2. The combination of the improved exporter and the black-box minimizer is much faster than minimizing a completely redundant projection because less axioms need to be tested. Nevertheless, there are still properties that can be used to efficiently remove redundancies from the projection:

Removing redundancies due to transitive roles

Theorem 14. *An explicit role assertion of an individual a, $R(a, c)$ is redundant in the projection $P(E, O_k, O)$, if there exists a role assertions $R(a, b)$ and $R(b, c)$ in $P(E, O_k, O) \cup O$ and R is a transitive role.*

Proof. The proof is a direct consequence of the definition of transitivity.

We will now show as one example why redundant role assertions that are induced by transitive roles cannot be avoided by an incremental export algorithm.

Theorem 15. *The reduction of transitive role assertions cannot be done incrementally during the export.*

Proof. We provide the proof using a counter-example: We assume that we first add the axioms to an individual $R(a, c)$ and $R(a, b)$. Unfortunately we did not yet export the individual b. Therefore, we cannot find the later inserted axiom $R(b, c)$ which would render $R(a, c)$ redundant in the incrementally produced projection. We could still change the order of exports and export c and b first. This could solve the example problem. But it is not a general solution because we may have a cycle in the property assertions that prevents us from finding a suitable order: Given the transitive and disjoint roles R and S and the axioms $R(a, b)$, $R(b, c)$, $R(c, d)$, $R(a, c)$, $R(a, d)$, $R(b, d)$ and $S(d, c)$, $S(c, b)$, $S(b, a)$, $S(d, b)$, $S(d, a)$, $S(c, a)$. In order to eliminate the redundancies of R we would need to begin the incremental export with d, in order to eliminate those of S we would need to start with a, which is impossible for an incremental export.

According to the theorem we cannot guarantee that the transitive role assertions are redundancy free when the check is done incrementally during the creation of the projection. In order to remove redundancies that are induced by transitive properties we use a simple nested loop algorithm that iterates over all transitive properties and all pairs of individuals in order to compute the transitive reduction [1] of the property assertions.

Another source of redundancies that cannot incrementally be handled during the export are redundancies that occur due to (global) ranges of properties (roles):

Removing redundancies due to ranges of roles

Theorem 16. *An explicit type assertion of an individual a, $C(a)$ is redundant in the projection $P(E, O_k, O)$ if there exists a role assertion $R(b, a)$ in $P(E, O_k, O) \cup O$ and C is equivalent to the range of R.*

Proof. The range C of a role R is defined in DL as: $\top \sqsubseteq \forall R.C$. Consequently it is redundant to specifically state that an instance a that is the target of a role assertion $R(x, a)$ is of the type C.

A minimizing algorithm can iterate over all role assertions and remove corresponding type assertions of the individuals that are in the range of the property assertions. A similar behavior exists for local ranges of a property:

Theorem 17. *An explicit type assertion of an individual a, $C(a)$ is redundant in the projection $P(E, O_k, O)$ if there exists a role assertion $R(b, a)$ in $P(E, O_k, O) \cup O$ and there exists a qualified universal role restriction on the class of b of the form $b \sqsubseteq \forall R.C$*

Proof. The proof is in analogy to the one of theorem 16 with the only difference that instead of the top class a specific class is used.

This property can easily be integrated into the proposed algorithm for the removal of redundancies that are induced by global ranges by additionally querying universal restrictions on the current individual.

Theorem 18. *The reduction of type assertions that are redundant due to local or global property ranges cannot be realized incrementally.*

Proof. As shown in proof of theorem 15 a cyclic dependency between properties cannot be resolved in an incremental way. The proof for the removal of redundant type assertions due to local or global ranges can be done in analogy.

Removing redundant same individuals assertions

Theorem 19. *An explicit same individual assertion $a \approx c$ or $c \approx a$ in the projection $P(E, O_k, O)$ is redundant if there exists a functional role assertion $R(a, x)$ and $R(c, y) \in P(E, O_k, O) \cup O$ and $x \approx y$.*

Proof. The proof is the direct consequence of the definition of functional roles.

This property can be used to minimize the set of same individuals by nested loops over each functional property and each individual with each other individual.

Theorem 20. *The removal of redundant same individual assertions due to functional properties cannot be realized with an incremental projection algorithm.*

Proof. The incremental algorithm exports each complete individual in E and never removes already exported axioms. It will export the individuals beginning at i_1 and ending at i_n. Given the role assertion $R(i_n, i_1)$ and R is a functional property and $R(i_{n-1}, i_1)$ it could potentially not export the same individual assertion $i_{n-1} \approx i_n$ but it will export $i_{n-1} \approx i_n$ because, when processing i_{n-1} it has no knowledge about i_n.

4 Evaluation

The discussed properties of Abox projections can be used to efficiently reduce the number of exported axioms. We have focused on frequent reasons for large numbers of redundant axioms but did not intend to guarantee the general minimality for the efficient algorithms. For example we do not consider redundancies that are induced by data-type / data-value reasoning and we do not handle redundancies that are inducted by property chains yet.
We have implemented both naive algorithms and an efficient exporter that exploits the theorems of section 3.1 and an efficient pre-minimizer according to the theorems in section 3.2. All algorithms were implemented in Java using the

Manchester OWL API[2] together with the pellet[3] reasoner. The experiments were executed on a virtual machine with dedicated 8GB of RAM and two dedicated CPU cores (AMD Opteron 2382 @ 2,61 Ghz). The aim of the evaluation was to evaluate the applicability of the proposed algorithms with regard to the performance (exported instances/second) and the degree of minimality that can be reached by our implementation. We used different real-world and benchmarking ontologies with big Aboxes (LUBM[4], Financial[5], Restaurant[6] and a generated dataset of provenance data based on ICD-10[7]) to cover a broad spectrum of ontologies.

4.1 Experimental Results

Scalability with regard to Tbox and export size: Our first experiment is the export of randomly generated data of patients, their diagnosis and their medical doctors in analogy to the example in section 1. The private source data only refers to private concepts and roles that are an extension of the target ontology using a generated nested hierarchy of 10 subclasses / subroles. To evaluate the performance with regard to big Tboxes we used an OWL version of the ICD 10 disease classification for the diagnosis. ICD 10 is organized in different chapters. To evaluate the scalability with regard to the size of the Tbox we executed the tests with different numbers of chapters. The Abox for all tests has a total size of 15,143 instances. The results are shown in figure 1. It shows the number of exported instances per second (exp. scale!) for different numbers of exp. individuals (x-axis) and different Tbox-sizes (A-F: 2907 classes, A -J: 4516 classes, A-O: 7058 classes and ICD all 14509 classes) and different algorithms. The efficient exporter together with the efficient minimizer achieves a performance between 3 and 36 instances per second. The results of the simple but full minimizer implementation that is executed on the result of the efficient exporter and efficient minimizer reaches a performance of 0.8 and 0.05 instances per second. Actually it turned out that for all experiments with the ICD 10 ontology the result of the efficient exporter with the efficient minimizer was already minimal. The experiments show that the efficient exporter reaches a good and nearly linear performance with regard to the number of exported instances. Only the startup costs make the export of very small numbers of instances inefficient. In contrast the full minimizer shows an exponential decrease in speed when the number of exported instances growth. We have also partly evaluated the naive exporter with the full minimizer. While the naive exporter performs better than the efficient exporter with regard to inst./sec. the minimizer gets much slower when executed on a large set of axioms.

[2] `http://owlapi.sourceforge.net`
[3] `http://clarkparsia.com/pellet`
[4] `http://swat.cse.lehigh.edu/projects/lubm/`
[5] `http://www.cs.put.poznan.pl/alawrynowicz/semintec.htm`
[6] `https://files.ifi.uzh.ch/ddis/oldweb/ddis/fileadmin/ont/nli/restaurant.owl`
[7] `https://dkm.fbk.eu/index.php/ICD-10\_Ontology`

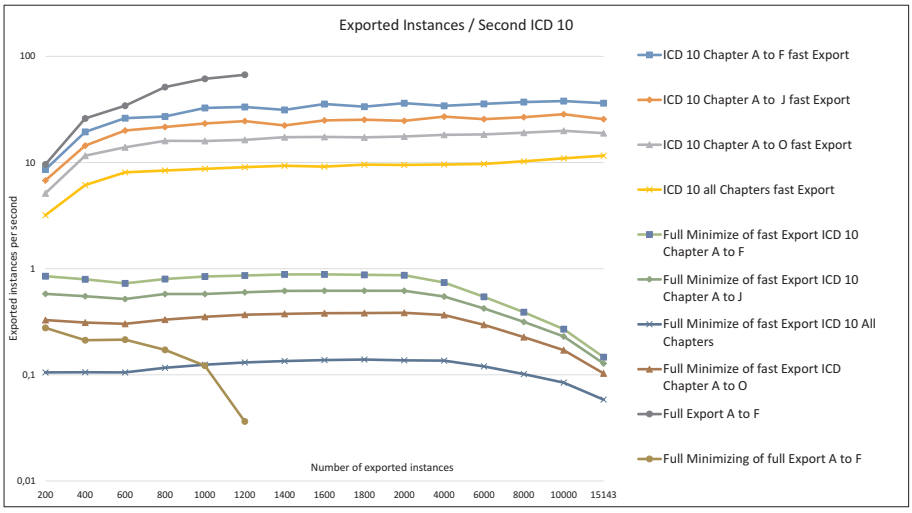

Fig. 1. Instances per second ICD10 with various Tbox sizes.

Export performance and degree of redundancy with various benchmarking ontologies: The second experiment compares the number of redundant axioms and the exported axioms per second of our efficient algorithm with the naive implementation. We have used typical benchmarking ontologies with large numbers of instances (LUBM (1 University, 1609 instances), Financial (17941 instances), Restaurant (9748 instances) and the ICD 10 testcase from the last experiment (15092 instances)). All ontologies were extended with a hierarchy of 10 subclasses / subroles and all instances were rewritten to only refer to the extended (private) classes and roles. We have always exported the whole Abox to a target ontology with an initially empty Abox. The number of exported axioms of our efficient exporter in comparison to the naive exporter for the different datasets is shown in figure 3. It is obvious that the structure of the ontology has a large impact on the number of redundant axioms that can be saved. The highest number of savings could be achieved in the ICD 10 test case due to the large Tbox. The redundancy of the restaurant ontology is also big because many redundancies are caused by transitive and inverse roles. The efficient exporter creates an export set that is around 70% smaller in comparison to the full export. In case of the financial ontology the number of exported axioms is still around 50% smaller. In case of the LUBM dataset the efficient exporter saves around 35% of the axioms of the full export.
The reduction of the number of axioms has also consequences for the file size of the export depending on the serialization method (we use the RDF/XML serialization). In the ICD 10 test case, the fully redundant export file (only Abox) has a size of 22.4 MB. In comparison the minimized set has a size of 7.58 MB. Therefore, it is around three times smaller. The same ratio holds, when both files are zipped (647 KB vs. 211 KB). The reduced size has also consequences

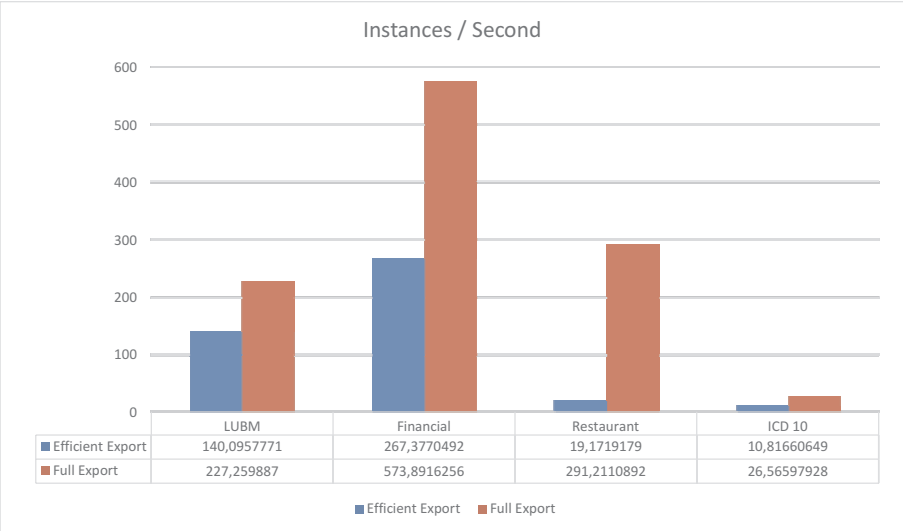

Fig. 2. Exported instances per second full vs. efficient export.

for the time that is required to load the ontology into the target system. In an average of 100 experiments the time to load the minimized Abox was 0.8 seconds in comparison to 2.1 seconds for the non minimized version. Therefore, the minimized version can be loaded 2.6 times faster than the full export.

Of course the efficient export requires more time to compute the output. The comparison of both approaches with regard to exported instances per second is shown in figure 2. While for LUBM, ICD 10 and Financial the number of instances per second of the efficient exporter is moderately lower (LUBM: 38%, ICD 10: 59%, Financial: 53%) the performance for the Restaurant data-set is 93% lower. This limited performance for the restaurant ontology is due to the fact that the restaurant ontology heavily uses inverse and transitive properties. All values for instances per second include the time for the efficient export and the post-minimizer. The amount of time for the post-minimizer in comparison to the total time was 17% for LUBM, 93% for Restaurant, 40% for Financial, and 5% for ICD-10.

Degree of minimality We have executed the black-box minimizer on all results to check the degree of minimality that could be achieved by our solution. In fact we produced minimal results for the ICD 10 and Restaurant datasets. For the LUBM benchmark the efficient exporter and minimizer did not eliminate one redundant axiom of 8000 axioms. In case of the Financial dataset our efficient algorithms did not eliminate around 50 redundant axioms of 50000. The cause of the redundant axioms are redundant type assertion due to defined (top-level) classes. In this case it can be deduced by the properties/roles of an instance that it has a specific type. Therefore, an additional assertion to this type is redundant. While one could argue that this is a major shortcoming of our approach it does

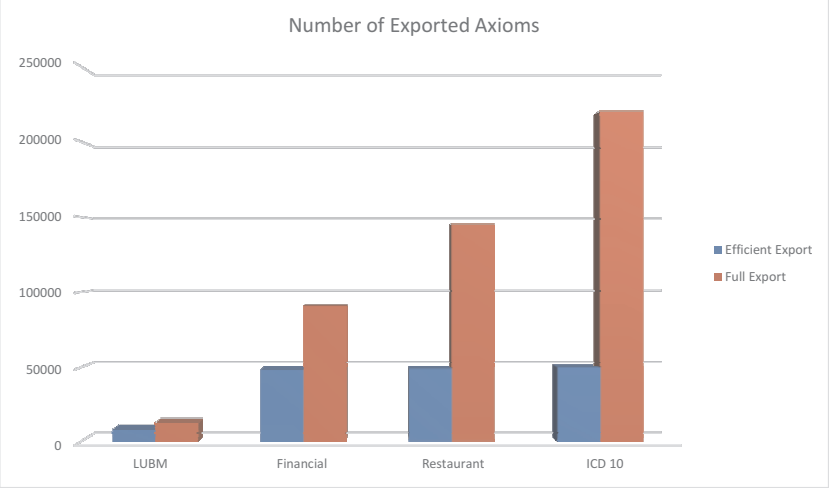

Fig. 3. Number of exported axioms full vs. efficient export.

not really result in big numbers of redundant axioms (1 of 8000 in case of LUBM and 50 of 50000 in case of financial). The reason for the still very low degree of redundancy is that in many cases our approach can still reach minimality when defined classes are used and the definition also addresses other classes in the hierarchy. For example a type assertion to the defined class *youngperson (person and hasAge* $\leq$ *20)* will not be redundant, for a *person* of the age of 10 because a *youngperson* is a subclass of *person*. Other redundancies that we do currently not address are redundancies by OWL 2 object property chains. However they can potentially be addressed in analogy to the processing of redundancies due to transitive properties.

Different Individuals Axioms Ontologies, where all sister classes are defined to be disjoint result in individuals with a very large set of different individuals assertions. Those are typically all redundant because they are implicitly defined by the Tbox of the shared ontology. A typical scenario is that for each individual, all other individuals are different. Even only querying all those different from axioms for each individual using the reasoner in the naive exporter is unfeasible for larger sets of instances. Therefore, for all tests with large Aboxes we only exported explicitly defined disjointness axioms.

5 Related Work

There exists numerous upper ontologies [15] and domain ontologies for nearly every domain. To our surprise methods to efficiently project instance data from private ontologies to corresponding domain- and upper ontologies have not been addressed so far. Related to our research are works for ontology translation such

as [6, 20]. In contrast to our scenario of projecting private instances to upper ontology instances the goal of ontology translation is to translate between arbitrary ontologies. The approach in [6] follows a similar idea as our approach: First different ontologies are mapped and then merged to one ontology. In a second step the data is projected from a source ontology to a target ontology. However, the setting is different because the translation between arbitrary ontologies requires more expressive mappings (such as translation functions). As a consequence the approach [6] translates the source ontologies to a more expressive language (first order logics), applies specific reasoning methods for translation and lowers the data back to the target ontology. In our setting of the translation to domain/upper ontologies case we can stay on the same (tractable) logical representation as the ontology itself and use any OWL reasoner. Another major difference is that our aim was to achieve minimal results, which, is only partly addressed in [6]. The ontology translation approach in [20] is based on distributed description logics, where different ontologies are mapped with specific bridging axioms. Similar to [6] it allows to reason over a merged ontology but it avoids to use first order logics as an intermediate format. The approach does not care for the miniamlity of the results and requires a specific reasoner. A major difference to the ontology translation approaches is that in our scenario the merged ontology is already given and is based on standard description logics. We are actually interested in extracting specific sets of entailments. Different approaches for the extraction of entailments are discussed in [3]. However, the goal is to construct sets that are interesting for the user for debugging purposes.

Another related field are approaches for modular ontologies [23]. In particular module extraction [16, 13, 5] is closely related. The idea is to extract a (minimal) sub-ontology for a specific signature from an ontology that preserves the entailments of the input ontology regarding the given signature. Therefore, module extraction solves a more general problem than our problem definition. It was shown in [5] that module extraction is undecidable for OWL DL. Therefore, current implementations use some approximations or impose additional constraints. Since the goal is to create sub-ontologies the generated modules also include additional concepts of the input ontology that describe the input signature but that are not elements of it and the input signature is typically limited to concepts. In contrast we are only interested in Abox axioms that can be expressed using the domain ontology and private concepts should never be part of the export due to privacy requirements.

Finally Abox abduction [18, 12] is somehow related to our approach. Reasoning methods for Abox abduction allow to answer queries like: Provide all minimal sets of assertions that must exist for a specific observation to hold. This can potentially also be used for minimizing of Abox axioms. In this case simply all derived or explicit axioms of an individual are observations. However, this is actually a trivial case for Abox abduction. The methods are tailored for non trivial cases and require non standard reasoning facilities that require multiple transformation steps resulting in a limited performance for our scenario. Our solution is based on standard description logics reasoners that are used in the

information systems of the partners anyway. The same reasoners can be used to compute the set of instances and to actually perform the export. No additional transformations are required.

6 Conclusion

In this paper we have addressed the problem of exporting instances of a private ontology to instances of a shared upper / domain ontology. Methods to generate such an export either suffer from very high degrees of redundant axioms or of simply unfeasible performance. The major contribution of this paper is a new efficient method for exporting Abox axioms to an upper ontology which generates export sets with low redundancy. The method is based on an systematic investigation of properties of Abox projections that result in redundant axioms. Of course, it would be desirable to guarantee export sets without any redundancy, however, we were more concerned about the performance of the export procedures and were ready to accept low degrees of redundancies. On the one hand the experiments showed that the method we propose here achieves very low degrees of redundancy between 0% and 0.01% in standard benchmarking ontologies. On the other hand the achieved performance is very promising and allows an application in real world use-cases such as the export of detailed provenance information in the biobank domain.

Acknowledgements: We would like to thank the anonymous reviewers for their detailed and valuable comments.

References

1. A. V. Aho, M. R. Garey, and J. D. Ullman. The transitive reduction of a directed graph. *SIAM J. Comput.*, 1(2):131–137, 1972.
2. M. Asslaber, P. M. Abuja, K. Stark, J. Eder, H. Gottweis, M. Trauner, H. Samonigg, H.-J. Mischinger, W. Schippinger, A. Berghold, et al. The genome austria tissue bank (gatib). *Pathobiology*, 74(4):251–258, 2007.
3. S. Bail, B. Parsia, and U. Sattler. Extracting finite sets of entailments from owl ontologies. In R. Rosati, S. Rudolph, and M. Zakharyaschev, editors, *Description Logics*, volume 745 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2011.
4. P. Buneman, A. Chapman, and J. Cheney. Provenance management in curated databases. In *Proc. of the ACM International Conference on Management of Data (SIGMOD)*, pages 539–550, New York, NY, USA, 2006. ACM Press.
5. B. Cuenca Grau, I. Horrocks, Y. Kazakov, and U. Sattler. Modular reuse of ontologies: Theory and practice. *J ARTIF INTELL RES*, 31:273–318, 2008.
6. D. Dou, D. McDermott, and P. Qi. Journal on data semantics ii. chapter Ontology translation on the semantic web, pages 35–57. Springer-Verlag, Berlin, Heidelberg, 2005.
7. J. Eder, C. Dabringer, M. Schicho, and K. Stark. Information systems for federated biobanks. *T. Large-Scale Data- and Knowledge-Centered Systems*, 1:156–190, 2009.
8. J. Eder, H. Gottweis, and K. Zatloukal. IT Solutions for Privacy Protection in Biobanking. *Public Health Genomics*, 15(5):254–262, 2012.

9. A. Gupta, C. Condit, and X. Qian. Biodb: An ontology-enhanced information system for heterogeneous biological information. *Data Knowl. Eng.*, 69(11):1084–1102, 2010.
10. T. Hernandez and S. Kambhampati. Integration of biological sources: current systems and challenges ahead. *SIGMOD Rec.*, 33(3):51–60, Sept. 2004.
11. A. Kalyanpur, B. Parsia, M. Horridge, and E. Sirin. Finding all justifications of owl dl entailments. In *ISWC/ASWC*, pages 267–280, 2007.
12. S. Klarman, U. Endriss, and S. Schlobach. Abox abduction in the description logic alc. *J. Autom. Reason.*, 46(1):43–80, Jan. 2011.
13. B. Konev, C. Lutz, D. Walther, and F. Wolter. Semantic modularity and module extraction in description logics. In *Proc. of ECAI 2008*, pages 55–59, Amsterdam, The Netherlands, The Netherlands, 2008. IOS Press.
14. M. Krötzsch, F. Simancik, and I. Horrocks. A description logic primer. *CoRR*, abs/1201.4089, 2012.
15. V. Mascardi, V. Cordì, and P. Rosso. A comparison of upper ontologies. In M. Baldoni, A. Boccalatte, F. D. Paoli, M. Martelli, and V. Mascardi, editors, *WOA*, pages 55–64. Seneca Edizioni Torino, 2007.
16. N. Noy and M. Musen. Specifying ontology views by traversal. In e. A. McIlraith, editor, *The Semantic Web ISWC 2004*, volume 3298 of *Lecture Notes in Computer Science*, pages 713–725. Springer Berlin Heidelberg, 2004.
17. W. OWL Working Group. *OWL 2 Web Ontology Language: Document Overview*. W3C Recommendation, 27 October 2009. Available at `http://www.w3.org/TR/owl2-overview/`.
18. J. Z. Pan, J. Du, G. Qi, and Y.-D. Shen. Towards practical abox abduction in large description logic ontologies. *Int. J. Semant. Web Inf. Syst.*, 8(2):1–33, Apr. 2012.
19. S. Ram and J. Liu. A new perspective on semantics of data provenance. In J. Freire, P. Missier, and S. S. Sahoo, editors, *SWPM*, volume 526 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2009.
20. L. Serafini and A. Tamilin. Instance migration in heterogeneous ontology environments. In K. e. A. Aberer, editor, *The Semantic Web*, volume 4825 of *Lecture Notes in Computer Science*, pages 452–465. Springer Berlin Heidelberg, 2007.
21. A. P. Sheth and J. A. Larson. Federated database systems for managing distributed, heterogeneous, and autonomous databases. *ACM Comput. Surv.*, 22(3):183–236, 1990.
22. Y. Simmhan, B. Plale, and D. Gannon. A survey of data provenance in e-science. *SIGMOD Record*, 34(3):31–36, 2005.
23. H. Stuckenschmidt, C. Parent, and S. Spaccapietra, editors. *Modular Ontologies: Concepts, Theories and Techniques for Knowledge Modularization*, volume 5445 of *Lecture Notes in Computer Science*. Springer, Berlin, 2009.
24. L. Teppo, E. Pukkala, and M. Lehtonen. Data quality and quality control of a population-based cancer registry. experience in finland. *Acta Oncologica (Stockholm, Sweden)*, 33(4):365–369, 1994. PMID: 8018367.
25. H.-E. Wichmann, K. A. Kuhn, M. Waldenberger, D. Schmelcher, S. Schuffenhauer, T. Meitinger, S. H. Wurst, G. Lamla, I. Fortier, P. R. Burton, et al. Comprehensive catalog of european biobanks. *Nature biotechnology*, 29(9):795–797, 2011.
26. M. Yuille, G.-J. B. van Ommen, C. Brchot, A. Cambon-Thomsen, G. Dagher, U. Landegren, J.-E. Litton, M. Pasterk, L. Peltonen, M. Taussig, H.-E. Wichmann, and K. Zatloukal. Biobanking for europe. *Briefings in Bioinformatics*, 9(1):14–24, 2008.