Reducing the Inferred Type Statements with Individual Grouping Constructs

Övünç Öztürk, Tuğba Özacar, and Murat Osman Ünalır

Department of Computer Engineering,
Ege University
Bornova, 35100, Izmir, Turkey
{ovunc.ozturk, tugba.ozacar, murat.osman.unalir}@ege.edu.tr

Abstract. A common approach for reasoning is to compute the deductive closure of an ontology using the rules specified and to work on the closure at query time. This approach reduces the run time complexity but increases the space requirements. The main reason of this increase is the type and subclass statements in the ontology. Type statements show a significant percentage in most ontologies. Since subclass is a transitive property, derivation of other statements, in particular type statements relying on it, gives rise to cyclic repetition and an excess of inferred type statements. In brief, a major part of closure computation is deriving the type statements relying on subclass statements. In this paper, we propose a syntactic transformation that is based on novel individual grouping constructs. This transformation reduces the number of inferred type statements relying on subclass relations. Thus, the space requirement of reasoning is reduced without affecting the soundness and the completeness.

1 Introduction

Semantic web applications will require multiple large ontologies for querying [1]. Although current reasoners are capable of schema reasoning with these real world ontologies, they break down for reasoning and retrieving the individuals in an ontology when the number of instances becomes large [2] [3]. It is a common and space consuming approach to compute the deductive closure of these large ontologies and to work on the closure at query time. This approach is known as offline reasoning that reduces run time complexity but increases space requirements.

Present work focusses at query-rewriting techniques. For example, [4] presents the *true RDF processor*. This processor provides *a slot access function* which allows the underlying graph to be viewed as if its RDFS-closure had been generated. This model has problems in dealing with domain and range restrictions and does not cover the complete RDF semantics. Another query rewriting approach for RDF reasoning [5], computes a small part of the implied statements offline thereby reducing space requirements, upload time and maintenance overhead. The computed fragment is chosen in such a way that the problem of inferring implied statements at run time can be reduced to a simple query rewriting. In

I. Cruz et al. (Eds.): ISWC 2006, LNCS 4273, pp. 573–582, 2006.

contrast to [4], [5] covers the complete RDF semantics by precomputing the transitive closure of hierarchical relations. These approaches do not compute complete closure offline thus they require online reasoning, which increases the runtime complexity.

In this paper, we propose a model that introduces novel individual grouping constructs for reducing the number of inferred type statements. The major problem with reasoning about individuals is deriving type statements relying on subclass relations between the classes. Our approach solves this problem with only a syntactic transformation. Instead of relating the class with each of its instances, this transformation relates the class with a group of sets where each set is a partial extension of that class. The advantages of our approach are listed below:

- covers complete semantics of RDF-based languages, since it is only a syntactic transformation
- does not affect the soundness and the completeness of the reasoning
- does not require query-rewriting and online reasoning
- does not require online reasoning

The rest of the paper is organized as follows: Section 2 introduces the model in detail and its subsections describe the transformations required by the model. Section 3 formulates the utilization rate of the model and evaluates utilization with large scale ontologies by means of ontology metrics in [6]. Finally Section 4 concludes the paper with an outline of some potential future research.

2 A Model with Individual Grouping Constructs

In this paper we propose a model that introduces individual grouping constructs for reducing the number of inferred type statements without affecting the soundness and the completeness of the reasoning.

An individual grouping construct is a partial extension of a class. The union of these partial extensions defines the exact list of class individuals. An individual of a class may be in one or more partial extensions of that class. These partial extensions are called *subExtension*s. A *subExtension* is related to one or more classes via *hasSubExt* predicate and it is related to individuals of the class with *contains* predicate. A *subExtension* is *direct* or *inherited* according to its relation with the class.

All individuals of a class, except the ones derived through a subclass relation, are added to the *directSubExtension* of that class. Each anonymous and non-anonymous class in the ontology has zero or one *directSubExtension*. Any class that has at least one individual has a *directSubExtension*. A class is related with its *directSubExtension* with *hasDirectSubExt* predicate. A *subExtension* can be related to one and only one class via *hasDirectSubExt* predicate.

An *inheritedSubExtension* of a class is the *directSubExtension* of a subclass of that class. In other words, an *inheritedSubExtension* holds the class individuals that are inherited to that class from one of its subclasses. A class is related to one or more *inheritedSubExtension*s with *hasInheritedSubExt* predicate. A

subExtension is related to one or more classes via an inferred *hasInheritedSubExt* relation. Note that, since every class is a subclass of itself, the *directSubExtension* of the class is also one of its *inheritedSubExtension*s. Thus every *subExtension* of a class is also an *inheritedSubExtension* of that class.

In our model the ontology is syntactically transformed to a semantically equivalent ontology with newly introduced individual grouping constructs. In order to preserve the completeness and the soundness of the reasoning, rules and queries are transformed in needs of these grouping constructs. These transformations [1] are described in the following subsections.

2.1 Triple Transformation

Let O be the set of triples in the ontology to be transformed, t be a type statement with subject I and object C then t is replaced with triple set S where;

$$S = \begin{cases} \{contains(E, I)\}, & \text{if } hasDirectSubExt(C, E) \in O \\ \{contains(E, I), hasDirectSubExt(C, E)\}, & \text{otherwise.} \end{cases}$$

It is worth to note that the same replacement is done whenever a type statement is added to or removed from ontology.

2.2 Rule/Query Transformation

Transforming rules is necessary to derive new statements with the transformed ontology without changing the completeness and the soundness of the reasoning. After the transformation, some rules have a form like $head :- body1 \vee body2$. These rules can be transformed easily via the following Lloyd-Topor transformation [7]:

$$head :- body1 \vee body2 \Rightarrow head :- body1 \text{ and } head :- body2$$

Note that it is necessary to extend the rule set to include the following rule, which specifies that every *directSubExtension* and *inheritedSubExtension* of a class is also a *subExtension* of that class:

$$\frac{hasDirectSubExt(B,E) \vee hasInheritedSubExt(B,E)}{hasSubExt(B,E)}$$

The following is *the standard rule transformation*, which is valid for all rules other than *RDF Rule 7*:

- A rule having a body condition with a type predicate is transformed in the following way :

$$\frac{\ldots \wedge type(U,B) \wedge \ldots}{h_1} \Rightarrow \frac{\ldots \wedge hasSubExt(B,E) \wedge contains(E,U) \wedge \ldots}{h_1}$$

- A rule having a head condition with a type predicate is transformed in the following way :

$$\frac{b_1 \wedge \ldots \wedge b_n}{type(U,B)} \Rightarrow \frac{b_1 \wedge \ldots \wedge b_n \wedge hasDirectSubExt(B,E)}{contains(E,U)}$$

[1] The transformation overhead is small enough to be considered negligible.

RDF Rule 7 computes the type statements relying on subclass relations. In order to reduce the number of inferred type statements, this rule is transformed in a different manner than the *standart rule transformation*. Instead of deriving a type relation between every individual of subclass and its superclass (Fig. 1a), *RDF Rule 7* simply derives only one relation with predicate *hasInheritedSubExt* between the *directSubExtension* and each *inheritedSubExtension* of subclass and its superclass (Fig. 1b). Table 1 demonstrates the transformation of some RDF rules from RDF Semantics [8], including *RDF Rule 7*.

Table 1. Transformation of RDF Rules

Rule No	RDFS Rule	Transformed Rule
(1)	$\frac{A(X,Y)}{type(A,Property)}$	$\frac{A(X,Y) \wedge hasDirectSubExt(Property,E)}{contains(E,A)}$
(2)	$\frac{domain(A,X) \wedge A(U,Y)}{type(U,X)}$	$\frac{domain(A,X) \wedge A(U,Y) \wedge hasDirectSubExt(X,E)}{contains(E,U)}$
(3)	$\frac{range(A,X) \wedge A(Y,V)}{type(V,X)}$	$\frac{range(A,X) \wedge A(Y,V) \wedge hasDirectSubExt(X,E)}{contains(E,V)}$
(4)	$\frac{A(U,B) \vee A(B,U)}{type(U,Resource)}$	$\frac{(A(U,B) \vee A(B,U)) \wedge hasDirectSubExt(Resource,E)}{contains(E,U)}$
(5)	$\frac{type(U,Property)}{subPropertyOf(U,U)}$	$\frac{hasSubExt(Property,E) \wedge contains(E,U)}{subPropertyOf(U,U)}$
(6)	$\frac{type(U,Class)}{subClassOf(U,Resource)}$	$\frac{hasSubExt(Class,E) \wedge contains(E,U)}{subClassOf(U,Resource)}$
(7)	$\frac{subClassOf(U,X) \wedge type(V,U)}{type(V,X)}$	$\frac{subClassOf(U,X) \wedge hasSubExt(U,E)}{hasInheritedSubExt(X,E)}$
(8)	$\frac{type(U,Class)}{subClassOf(U,U)}$	$\frac{hasDirectSubExt(Class,E) \wedge contains(E,U)}{subClassOf(U,U)}$

A query is a rule without a head thus the standard rule transformation is also applicable to queries. The queries are transformed in the following way where h_1 is an empty set:

$$\frac{...\wedge type(U,B)\wedge...}{h_1} \Rightarrow \frac{...\wedge hasSubExt(B,E)\wedge contains(E,U)\wedge...}{h_1}$$

It is also important to represent the answer in the standard way. Let S be the set of all triples in the answer set then replace the following triple sets with triple $type(U, B)$:

- $\{hasDirectSubExt(B, E), contains(E, U)\}$
- $\{hasInheritedSubExt(B, E), contains(E, U)\}$
- $\{hasSubExt(B, E), contains(E, U)\}$

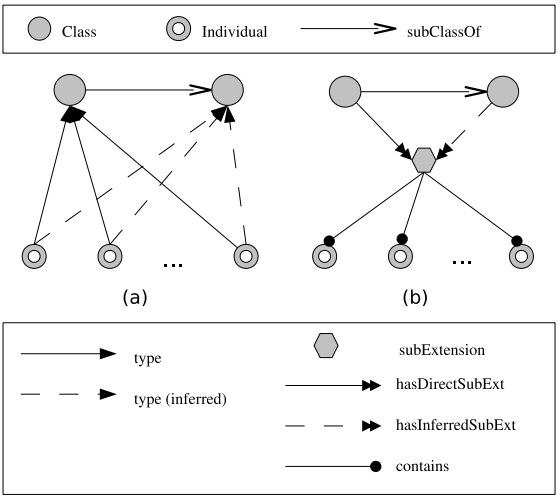

Fig. 1. Reducing the inferred type statements relying on subclass relations

3 Analyzing the Utilization of the Model

3.1 Computation of the Utilization Rate

Computation of the utilization rate reveals whether the model is valuable to use with a specific ontology. In order to calculate the utilization rate, the number of type statements in the ontology and the number of subclass relations in the closure of the ontology must be known. Since it is necessary to know only the number of subclass statements in the closure, it is enough to find the closure of the schema of the ontology.

The following formula calculates the decrease in the inferred triple count by applying the proposed model. Let Δ be the decrease in the inferred triple count by applying the model. The formula finds Δ by calculating the difference between the decrease in inferred type statements and the newly added or inferred triples related with grouping constructs. Before applying the model, every individual of a subclass is added to the individuals of its every superclass if the superclass does not have this individual. The number of these additions is $\mathcal{T}$. This value is the decrease in inferred type statements. In order to calculate the exact decrease in the inferred statements by applying the model, we have to minus the number of newly added *hasDirectSubExt*, *hasInferredSubExt* and *hasSubExt* statements from this value. Let $\mathcal{D}$ be the number of newly added *hasDirectSubExt* statements, $\mathcal{I}$ be the number of inferred *hasInferredSubExt* statements and $\mathcal{S}$ be the number of *hasSubExt* statements, then;

$$\Delta = \mathcal{T} - (\mathcal{I} + \mathcal{D} + \mathcal{S}) \tag{1}$$

Let n be the number of classes in the ontology, $\mathcal{E}_{C_x}$ be the set of all explicit individuals of class C_x where $1 \leq x \leq n$, $|X|$ be the number of elements in set

X, $\setminus$ be the set minus operator and $\delta\ (C_x, C_y)$ be a function which returns 1 if C_x is an explicit or implicit subclass of C_y. Otherwise the returned value is 0. And let $\phi\ (C_x)$ be a function, which computes the number of elements in the set-theoretic difference of $\mathcal{U}$ and C_x, where $\mathcal{U}$ is the union of all explicit and implicit subclasses of C_x.

$$\mathcal{T} = \sum_{i=1}^{n} \Phi(C_i) \tag{2}$$

$$\Phi(C_x) = |\bigcup_{i=1}^{n} \sigma(C_i, C_x)|$$

$$\sigma(C_x, C_y) = \begin{cases} \mathcal{E}_{C_x} \setminus \mathcal{E}_{C_y}, \text{ if } C_x \text{ is a subclass of } C_y \\ \\ \emptyset, \qquad\qquad \text{otherwise.} \end{cases}$$

$$\mathcal{I} = \sum_{i=1}^{n} \sum_{j=1}^{n} \delta\ (C_i, C_j) \tag{3}$$

$$\delta\ (C_x, C_y) = \begin{cases} 1, \text{ if } C_x \text{ is a subclass of } C_y \\ \\ 0, \text{ otherwise.} \end{cases}$$

Let $\gamma\ (C_x)$ be a function which returns 0, if C_x is an empty set. Otherwise the returned value is 1. Each non-empty class in the ontology has one and only one *directSubExtension*, thus

$$\mathcal{D} = \sum_{i=1}^{n} \gamma\ (C_i) \tag{4}$$

$$\gamma(C_x) = \begin{cases} 0, \text{ if } \mathcal{E}_{C_x} = \emptyset \\ \\ 1, \text{ otherwise.} \end{cases}$$

One and only one *subExtension* of a class is related to that class with *hasDirectSubExt* predicate. All the other *subExtension*s are related to the class with *hasInheritedSubExt* predicate. Since every class is a subclass of itself, the only *subExtension* that is related to the class with *hasDirectSubExt* predicate is also related to that class with *hasInheritedSubExt* predicate. Thus, all *subExtension*s of a class are also related to the class with *hasInheritedSubExt* predicate. Then, we can conclude that:

$$\mathcal{S} = \mathcal{I} \tag{5}$$

The utilization rate is the ratio between the reduction in inferred statements and the number of statements in the closure without implementing the model. Let t be the number of statements in the ontology before inference, ρ be the number of inferred statements without implementing the model, then the utilization rate of the model $\mathcal{U}$ is;

$$\mathcal{U} = 1 - \frac{\Delta}{t + \rho}$$

If the model is useful and reduces the number of inferred statements, the utilization rate is in the interval of $(0, 1]$. The closer the value is to 0, the higher the utilization of the model. If the utilization rate is equal to 1, then applying the model shows no impact. If the utilization rate is greater than 1, then applying the model shows a negative impact (i.e., the number of inferred statements increases).

3.2 Estimation of the Utilization Rate Using Ontology Metrics

The model shows a negative impact when the ontology has no subclass relation or the number of type statements in the ontology is less than the number of class definitions and subclass relations in the ontology. These two factors result in a relatively low number of inferred type statements in the closure of the ontology. In this case, the number of additional statements required by the technique is greater than the decrease in the number of inferred type statements.

Obviously, this case is not very common in real world ontologies and also is not meaningful from the perspective of knowledge representation. Table 2 depicts this issue by supplementing the number of subclass and type relations in representative ontologies. The common characteristics of ontologies in Table 2 are the reasonable amount of type and subclass statements and a significant increase in type statements in their corresponding closures [5]. Consequently, the model is valuable enough with these ontologies, but we enumerate the factors that increase the utilization rate in order to figure out when the model is applicable. These factors are:

- high number of subclass relations
- high number of individuals

Table 2. Behaviour of realistic data

	CIA WFB		TAP KB		SUMO		WordNet	
	orig.	closure	orig.	closure	orig.	closure	orig.	closure
type	376	4150	36988	191972	6709	18198	99653	277267
subClassOf	9	60	283	1491	3797	18439	78446	606418
other	25899	26077	71148	71253	8637	10584	295529	374032

In order to estimate whether the model is suitable for a particular ontology, the preceding factors can be associated with ontological metrics. Ontological metrics are used to assess the quality of an ontology. The quality of an ontology can be valuated in different aspects. These aspects are mainly grouped according to their relevance with ontology, they are related either with schema or with knowledge base, or both. Previous work in the literature is classified in [6], and one dimension of this classification is the relevance with ontology. In accordance with the classification, we decided to use metrics defined in OntoQA [6].

OntoQA metrics are mainly divided into schema metrics and instance metrics. Schema metrics are about the structure of the ontology. Instance metrics are about the population and its distribution over the classes. We selected two metrics from the OntoQA metric set: *inheritance richness* and *average population*. In [6], *inheritance richness* is defined as "the average number of subclasses per class", and *average population* is defined as "the number of instances of the knowledge base divided by the number of classes defined in the ontology schema". These metrics are proportional to the utilization of the model. Although an increase in average population results in an increase in the utilization of the model, it is necessary to note that the population intensifying in the leaf classes has a greater effect.

3.3 Evaluation of the Utilization Rate with Large Scale Ontologies

We also applied the model on a RETE based inference engine [9], with Lehigh University Benchmark (LUBM) in order to exemplify utilization of the model [10]. LUBM is developed to evaluate the inference and query performance of semantic web repositories over a large data set. The data set is generated according to one ontology, named *univ-bench*, using the synthetic data generation tool provided with the benchmark. LUBM can generate datasets with chosen size using the schema, for example LUBM(2,0) means the ontology includes two university instances with their corresponding dataset.

We conducted the test using four different rule sets. The first two sets are related with RDF semantics. The first one contains rules covering the semantics of *partial rdfs* [11], where some of the normative RDFS entailments are discharged for performance reasons. The second one is the transformed form of the previous rule set in needs of the proposed model. The remaining two sets are related with OWL semantics [12]. The first one contains rules covering the semantics of a DL that is between OWL Lite and OWL DL. This set contains 30 rules, which satisfy completeness and soundness of LUBM queries by 100 percent. Finally,

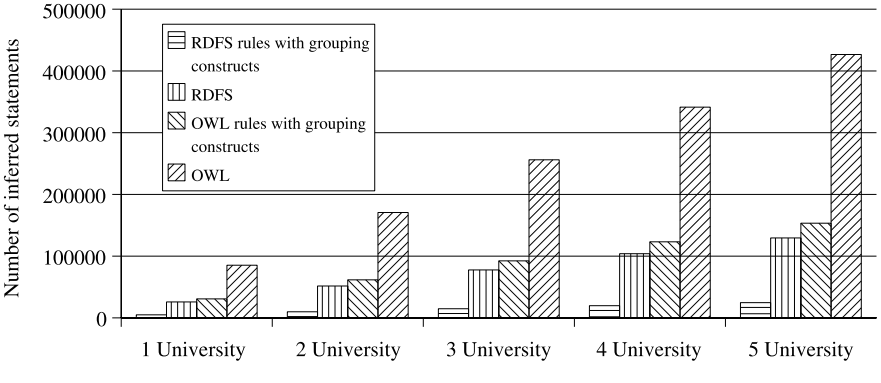

Fig. 2. Number of inferred statements in LUBM ontologies

the last rule set is the transformed form of the rule set having 30 OWL rules, in needs of the proposed model.

As can be seen from Figure 2, the model significantly reduces the number of inferred triples and also the number of total triples in the closure of the ontology. Therefore, the inferencing time of the ontology and the memory required to open the ontology decrease. The utilization is higher in RDFS level because in RDFS the ratio between inferred type statements and inferred statements is greater than in OWL. Also query time is not affected because there is no need to make backward chaining as in [5].

4 Conclusion and Future Work

This work is an attempt to reduce the number of inferred statements in the closure of web ontologies. It is just a syntactic transformation that does not change the soundness or the completeness of the reasoning. Instead of relating a class with each of its instances, this transformation relates the class with a group of sets where each set is a partial extension of that class.

The model is valuable with real world ontologies having a reasonable amount of individuals. This work formulates the utilization and also tries to associate it with ontology metrics in literature to estimate the utilization with a specific ontology. Since there is not enough work related with ontology metrics and the values of existent metrics with well known ontologies are not presented, we couldn't give a detailed work on the association of the utilization and the metrics. It will be an interesting future work to define the intervals of metrics in which the utilization is best.

Another potential future work is to reduce the semantic coverage of the model and specialize it to a specific ontology language, in order to gain additional utilization rate. This will be possible by defining not only individual grouping constructs, but also additional specific grouping constructs for a specific ontology language (e.g. grouping constructs specific for RDF).

References

1. Wielemaker, J., Schreiber, G., Wielinga, B.J.: Prolog-based infrastructure for rdf: Scalability and performance. In: International Semantic Web Conference. (2003) 644–658
2. Haarslev, V., Möller, R.: High performance reasoning with very large knowledge bases: A practical case study. In: IJCAI. (2001) 161–168
3. Horrocks, I., Li, L., Turi, D., Bechhofer, S.: The instance store: Dl reasoning with large numbers of individuals. In: Description Logics. (2004)
4. Lassila, O.: Taking the rdf model theory out for a spin. In: International Semantic Web Conference. (2002) 307–317
5. Stuckenschmidt, H., Broekstra, J.: Time - space trade-offs in scaling up rdf schema reasoning. In: WISE Workshops. (2005) 172–181

6. Tartir, S., Arpinar, I.B., Moore, M., Sheth, A.P., Aleman-Meza, B.: Ontoqa: Metric-based ontology quality analysis. In: Proceedings of IEEE ICDM 2005 Workshop on Knowledge Acquisition from Distributed, Autonomous, Semantically Heterogeneous Data and Knowledge Sources. (2005)
7. Lloyd, J.W.: Foundations of Logic Programming, 2nd Edition. Springer (1987)
8. Hayes, P.: Rdf semantics (2004)
9. Ünalir, M., Özacar, T., Öztürk, Ö.: Reordering query and rule patterns for query answering in a rete-based inference engine. In: WISE Workshops. (2005) 255–265
10. Guo, Y., Pan, Z., Heflin, J.: An evaluation of knowledge base systems for large owl datasets. In: International Semantic Web Conference. (2004) 274–288
11. Horst, H.J.: Combining rdf and part of owl with rules: Semantics, decidability, complexity. In: International Semantic Web Conference. (2005) 668–684
12. Patel-Schneider, P.F., Hayes, P., Horrocks, I.: Rdf semantics (2004)