Ontology Verbalization using Semantic-Refinement

Editor(s): Name Surname, University, Country
Solicited review(s): Name Surname, University, Country
Open review(s): Name Surname, University, Country

Vinu Ellampallil Venugopal * and P Sreenivasa Kumar
[a] *Computer Science and Communications Research Unit, University of Luxembourg, Belval, Luxembourg*
E-mail: vinu.venugopal@uni.lu
[b] *Department of Computer Science and Engineering, Indian Institute of Technology Madras, Chennai, India.*
E-mail: psk@cse.iitm.ac.in

Abstract. This paper presents an inference-based verbalization approach for OWL axioms that aims at removing redundancy to provide more helpful and more concise sentences to show formal facts to users. We focus on verbalization of OWL axioms about individual and (atomic) concepts as it is relevant in the context of validating the correctness of the formalized domain knowledge by domain experts. The approaches that are currently adopted in the existing verbalization tools generally consider all axioms that are associated with the individual (or the concept) under consideration and then translate them to corresponding natural language texts. Further refinement (mainly, grouping and aggregation) of these texts would be done at the natural language level to yield a more fluent and comprehensive form. However, we observed that human-understandability of such descriptions is affected by the presence of repetitions and redundancies which can be removed easily at the semantic level. We propose a novel technique called *semantic-refinement* which fulfills this requirement. This technique utilizes a predefined set of rules that are repeatedly applied over the restrictions that are associated with an individual (or a concept) in a meaning-preserving manner to get a refined set of restriction that can be verbalized to get a concise description. Our experiments on two ontologies show that semantic-refinement technique could significantly improve the readability of the natural language descriptions on comparing to those descriptions that are generated without employing the semantic-refinement. We have also tested the effectiveness and usefulness of the the generated descriptions in validating the correctness of ontologies and found that the proposed technique is indeed helpful in that context.

Keywords: Verbalization, Ontologies, Rule-based system

1. Introduction

Web Ontology Language (or Description Logic based) ontologies are knowledge representation structures which are based on decidable fragments of first order logic. They model domain knowledge in the form of logical axioms; so that an intelligent agent with the help of a reasoning system can make use of them for several applications. While ontologies are used primarily as a source of vocabulary for standardization and integration purposes, many applications also use them as a source of computable knowledge. Since the knowledge in the form of an ontology is inherently characterized by complex logical axioms, it is typically inaccessible for non-ontology community. This problem motivated researchers to work on natural language (NL) verbalization techniques for OWL ontologies. The existing approaches in this direction mainly strive for converting each logical construct into corresponding NL fragment, and result in methods which produce verbatim equivalents of Web Ontol-

*Corresponding author. E-mail: vinu.venugopal@uni.lu, mvsquare1729@gmail.com

ogy Language (OWL) constructs. One of the main and common drawback of these approaches is that, since the generated sentences are verbatim equivalent to the OWL statements, they are likely to have high amount of redundancy. As we show later with examples, it can be very annoying for a human reader (especially domain experts) to read and understand such sentences. Therefore, in this paper, we explore techniques which can generate NL sentences that do not have redundancies and are semantically equivalent to their OWL counterparts.

We will closely look at the problem of verbalization of OWL ontologies from the perspective of using the generated descriptions for validating the correctness of the formalized knowledge. Typically, ontologies are developed by a group of knowledge engineers with the help of domain experts [4]. The domain experts provide the knowledge to be formalized and the engineers build the ontology out of it. Since an ontology development involves multiple parties (engineers and domain experts), the process usually follows an iterative model which aims at building the final model through a series of small steps and at the end of each iteration a particular preliminary version of the ontology is released [13].

As an ontology typically evolves over time with their different parts being developed and maintained in different contexts and separately from each other, which is a potential source for redundancy if such parts are merged [6]. Unless these merges/updates are carefully carried out, the quality of the ontology might degrade [5]. To prevent such quality depletion, usually an ontology development cycle is accompanied by a validation phase (to validate the correctness of the OWL statements), where both the knowledge engineers and domain experts meet to review the content of the ontology [14].

In a typical validation phase, new axioms are included or existing axioms are altered or removed, to maintain the correctness of the ontology. The conventional method for incorporating new axioms and validating the ontology involves a validity check by domain experts. Domain experts, who do the validity check, cannot be expected to be highly knowledgeable on formal methods and notations. For their convenience, the OWL axioms will have to be first converted into corresponding NL texts. Ontology verbalizers and ontology authoring tools such as ACE [12], NaturalOWL [1] and SWAT Tools [18], can be utilized for generating controlled natural language (CNL) descriptions of OWL statements. Restricting our attention to description of individuals and atomic concepts, we find that the approach currently followed in the available tools is that of determining the set of all logical conditions that are satisfied by the given individual/concept name and translate these conditions verbatim into corresponding NL descriptions. But the verbatim fidelity of such descriptions to the underlying OWL statements, makes them a poor choice for ontology validation. This is because, the descriptions will be confusing to a person who is not familiar with formal constructs, and it will be difficult to correctly understand the meaning from such descriptions. This issue had been previously reported in papers such as [16,18], where the authors tried to overcome the issue by applying operations such as grouping and aggregation on the verbalized text. But, since the issue had been treated at the NL text level, the opportunity for a logical-level refinement of the OWL statements to generate a more meaningful and human-understandable representation has been ignored.

For example, consider the following logical axioms (from People & Pets ontology[1]) represented in the description logic (DL) notation.

1. $\texttt{Cat\_Owner} \sqsubseteq \texttt{Person} \sqcap \texttt{Owner} \sqcap \exists\texttt{hasPet.Animal} \sqcap \exists\texttt{hasPet.Cat}$
2. $\texttt{Cat\_Owner(sam)}$
3. $\texttt{Cat} \sqsubseteq \texttt{Animal}$

The different variants of the CNL sentences corresponding to the individual `sam` are as follows:

– *A cat-owner is a person[2]. A cat-owner is an owner. A cat-owner has as pet an animal. A cat-owner has as pet a cat. Sam is a cat-owner. All cats are animals.*
  or (with grouping and aggregation)
– *A cat-owner is a person and an owner . A cat-owner is all of the following: something that has pet an animal, and something that has pet a cat; Example: sam. All cats are animals.*

As can be easily noted, these descriptions have redundant information and attempting verbatim equivalence to DL constructs has resulted in this situation. There are different types of redundancies one can observe. The obvious type is repetition of linguistically similar texts; for example, a cat-owner is an owner. Another type includes those generic restrictions which

[1] http://www.cs.man.ac.uk/~horrocks/ISWC2003/Tutorial /people+pets.owl.rdf

[2] Other way of saying "All cat-owners are person"

can be logically inferred from more specific restrictions; for example, having said "A cat-owner has as pet a cat.", it is not necessary to say "A cat-owner has as pet an animal." This paper deals with removing redundancies of the latter kind. Furthermore, OWL axioms involving properties can be expected to be harder to understand, given that properties are logically more complex than individuals and concepts. Therefore the restriction to individuals and concepts is a serious one that limits the potential impact and usefulness.

In this paper, we introduce an approach for removing redundancies from the verbalized definitions of OWL entities, and to generate the so-called *redundancy-free* representations/descriptions. We propose a technique called *semantic-level refinement* (or simply *semantic-refinement*) that helps in removing the redundant (portion of the) restrictions and generating a more semantically comprehensive description of the entity. From an application point of view, in this paper, we particularly focus on generating NL descriptions of *individuals* and *concepts* for validating ontologies which follow $\mathcal{SHIQ}$ description logic.

Our proposed approach generates NL descriptions of individuals and concepts by giving importance to the semantic conciseness of the content. If we revisit our previous example, we expect our approach to produce a text similar to: *Sam: is a cat-owner having at least one cat as pet*; such that the redundant portion of the text *has as pet an animal* (since it clearly follows from *having at least one cat as pet*) is removed.

This paper is arranged as follows: Section 3 and 4 discuss the preliminaries for understanding the work and, newly introduced terminologies in the paper respectively. In Section 5 we elaborate an approach for generating definitions (in the form of logical expressions) of ontology individuals and concepts, and a rule-based method for removing redundancies from the definitions. Section 6 explains the process that we have followed for generating NL sentences from the logical expressions.

In Section 7, the empirical evaluation section, we seek to validate the following two propositions using case studies and using statistical significance tests. Firstly, logical-level removal of redundancies and repetitions can significantly improve the understandability of the domain knowledge when expressed in a NL, for domain experts. Secondly, NL definitions of individuals of an ontology can be effectively used for validating the ontology.

2. Related Work

Over the last decade, several CNLs such as Attempto Controlled English (ACE) [12,11], Ordnance Survey[3]'s Rabbit (Rabbit) [7], and Sydney OWL Syntax (SOS) [3], have been specifically designed or have been adapted for ontology language OWL. All these languages are meant to make the interactions with formal ontological statements easier and faster for users who are unfamiliar with formal notations. Unlike the other languages [8,10,1] that have been suggested to represent OWL in controlled English, these CNLs are designed to have formal language semantics and bidirectional mapping between NL fragments and OWL constructs. Even though these formal language semantics and bidirectional mapping enable a formal check to determine if the resulting NL expressions are unambiguous, they can result in generating a collection of unordered sentences that are difficult to comprehend.

To use these CNLs as a means for ontology authoring and for knowledge validation purposes, the verbalized texts have to be properly organized. A detailed comparison of the systems that do such text organization is given in [16]. Among such systems, SWAT tools[4] are one of the recent and prominent tools which use standard techniques from computational linguistics to make the verbalized text more readable. They have tried to give better clarity to the generated text by grouping, aggregation and elision. The Semantic Web Authoring (SWAT) NL verbalization tools have given much importance to the fluency of the verbalized sentences [18], rather than removing redundancies from their logical forms, hence have deficiencies in interpreting the ontology contents.

A notion for removing redundancies from ontologies, similar to what we propose in this paper, was proposed first in [6]. However, they have look at redundancy in ontologies primarily from an ontology engineering and evolution point of view. A notion for removing redundancies in a verbalization perspective was first introduced in [17]. In that paper, the authors have clearly established the fact that omitting "obvious" axioms while verbalization leads to a better reading experience for a human. By "obvious" axioms the author means those axioms whose semantics are in some sense obvious for an average human reader. (For example, phrases such as "junior school" explicitly covey the meaning that a junior school is a school.)

[3]Great Britain's national mapping agency
[4]http://mcs.open.ac.uk/nlg/SWAT/

Table 1
The syntax and semantics of $\mathcal{SHIQ}$ concept types

Name	Syntax	Semantics
atomic concept	A	$A^{\mathcal{I}}$
top concept	$\top$	$\Delta^{\mathcal{I}}$
bottom concept	$\bot$	ϕ
negation	$\neg C$	$\Delta^{\mathcal{I}} \backslash C^{\mathcal{I}}$
conjunction	$C \sqcap D$	$C^{\mathcal{I}} \cap D^{\mathcal{I}}$
disjunction	$C \sqcup D$	$C^{\mathcal{I}} \cup D^{\mathcal{I}}$
existential restriction	$\exists R.C$	$\{ x \in \Delta^{\mathcal{I}} \mid \exists y.\langle x, y\rangle \in R^{\mathcal{I}} \wedge y \in C^{\mathcal{I}} \}$
universal restriction	$\forall R.C$	$\{ x \in \Delta^{\mathcal{I}} \mid \forall y.\langle x, y\rangle \in R^{\mathcal{I}} \Rightarrow y \in C^{\mathcal{I}} \}$
min cardinality	$\geq nR.C$	$\{ x \in \Delta^{\mathcal{I}} \mid \#\{y \mid \langle x, y\rangle \in R^{\mathcal{I}} \wedge y \in C^{\mathcal{I}}\} \geq n \}$
max cardinality	$\leq mR.C$	$\{ x \in \Delta^{\mathcal{I}} \mid \#\{y \mid \langle x, y\rangle \in R^{\mathcal{I}} \wedge y \in C^{\mathcal{I}}\} \leq m \}$

Table 2
The syntax and semantics of $\mathcal{SHIQ}$ ontology axioms

	Name	Syntax	Semantics
TBox	role hierarchy	$R \sqsubseteq S$	$R^{\mathcal{I}} \subseteq S^{\mathcal{I}}$
	role transitivity	Tran(R)	$R^{\mathcal{I}} \circ R^{\mathcal{I}} \subseteq R^{\mathcal{I}}$
	concept inclusion	$C \sqsubseteq D$	$C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$
	concept equality	$C \equiv D$	$C^{\mathcal{I}} = D^{\mathcal{I}}$
ABox	concept assertion	$C(a)$	$a^{\mathcal{I}} \in C^{\mathcal{I}}$
	role assertion	$R(a, b)$	$\langle a^{\mathcal{I}}, b^{\mathcal{I}}\rangle \in R^{\mathcal{I}}$
	inequality assertion	$a \not\approx b$	$a^{\mathcal{I}} \neq b^{\mathcal{I}}$

In our work, we, go further and, establish that more inference-based redundancy removal can be performed (similar to the notions proposed in [6]) than removing just the morphological variants of the entity names, for greatly improving the quality and understandability of verbalized text.

3. Preliminaries

3.1. $\mathcal{SHIQ}$ Ontologies

The description logic (DL) $\mathcal{SHIQ}$ is based on an extension of the well-known logic $\mathcal{ALC}$ [15], with added support for role hierarchies, inverse roles, transitive roles, and qualifying number restrictions [9].

We assume N_C and N_R as countably infinite disjoint sets of *atomic concepts* and *atomic roles* respectively. A $\mathcal{SHIQ}$ *role* is either $R \in N_R$ or an *inverse role* R^- with $R \in N_R$. To avoid considering roles such as $(R^-)^-$, we define a function Inv(.) which returns the inverse of a role: Inv$(R) = R^-$ and Inv$(R^-) = R$.

The set of concepts in $\mathcal{SHIQ}$ is recursively defined using the constructors in Table 1, where $A \in N_C, C, D$ are concepts, R, S are roles, and n, m are positive integers. A $\mathcal{SHIQ}$ based ontology — denoted as a pair $\mathcal{O} = (T, A)$, where T denotes terminological axioms (also known as TBox) and A represents assertional axioms (also known as ABox) — is a set of axioms of the type specified in Table 2. A role R in $\mathcal{O}$ is *transitive* if Tran$(R) \in \mathcal{O}$ or Tran$(R^-) \in \mathcal{O}$. Given an $\mathcal{O}$, let $\sqsubseteq_{\mathcal{O}}$ be the smallest transitive reflexive relation between roles R_1 and R_2, such that $R_1 \sqsubseteq R_2 \in \mathcal{O}$ implies $R_1 \sqsubseteq_{\mathcal{O}} R_2$ and $R_1^- \sqsubseteq_{\mathcal{O}} R_2^-$. For a $\mathcal{SHIQ}$ ontology $\mathcal{O}$, the role S in every concept of the form $\geq nS.C$ and $\leq mS.C$ in $\mathcal{O}$, should be *simple*, that is, $R \sqsubseteq_{\mathcal{O}} S$ holds for no transitive role R [2].

The semantics of $\mathcal{SHIQ}$ is defined using *interpretations*. An interpretation is a pair $\mathcal{I} = (\Delta^{\mathcal{I}}, .^{\mathcal{I}})$ where $\Delta^{\mathcal{I}}$ is a non-empty set called the *domain* of the interpretation and $.^{\mathcal{I}}$ is the *interpretation function*. The function $.^{\mathcal{I}}$ assigns a set $A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$ to every $A \in N_C$, and assigns a relation $r^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$ to every $r \in N_R$. The interpretation of the inverse role r^- is $(r^-)^{\mathcal{I}} := \{\langle x, y\rangle \mid \langle y, x\rangle \in r^{\mathcal{I}}\}$. The interpretation is extended to concepts and axioms according to the rightmost column of Table 1 and Table 2 respectively, where $\#X$ denotes the cardinality of the set X.

We write $\mathcal{I} \models \alpha$, if the interpretation $\mathcal{I}$ satisfies the axiom α (or α is *true* in $\mathcal{I}$). $\mathcal{I}$ is a *model* of an ontology $\mathcal{O}$ (written $\mathcal{I} \models \mathcal{O}$) if $\mathcal{I}$ satisfies every axiom in $\mathcal{O}$. If we say α is entailed by $\mathcal{O}$, or α is a *logical consequence* of $\mathcal{O}$ (written $\mathcal{O} \models \alpha$), then every model of $\mathcal{O}$ satisfies α. A concept C is *subsumed* by D w.r.t. $\mathcal{O}$ if $\mathcal{O} \models C \sqsubseteq D$, and C is *unsatisfiable* w.r.t. $\mathcal{O}$ if $\mathcal{O} \models C \sqsubseteq \bot$. *Classification* is the task of computing all subsumptions $A \sqsubseteq B$ between atomic concepts such that $A, B \in N_C$ and $\mathcal{O} \models A \sqsubseteq B$; similarly, *property classification* of $\mathcal{O}$ is the computation of all subsumptions between properties $R \sqsubseteq S$ such that $R, S \in N_R$ and $\mathcal{O} \models R \sqsubseteq S$.

3.2. Running Example

In this section we introduce an example ontology (called the academic (ACAD) ontology) which we follow throughout this chapter. We have formalized various concepts in academic domain in this ontology. The ontology is rather small, but serves the purpose well. The TBox and ABox of the ontology is given in Table 3 and 4 respectively.

Table 3

TBox of ACAD ontology

IITStudent	$\equiv$	Student $\sqcap$ $\forall$hasAdvisor.TeachingStaff $\sqcap$ $\exists$hasAdvisor.Professor $\sqcap$ $\exists$enrolledIn.IITProgramme
IIT_MS_Student	$\equiv$	IITStudent $\sqcap$ ≤ 1 hasAdvisor.TeachingStaff
IITPhdStudent	$\equiv$	IITStudent $\sqcap$ ≥ 2 hasAdvisor.TeachingStaff $\sqcap$ ≤ 1 hasAdvisor.Professor
Professor	$\sqsubseteq$	TeachingStaff
AssistantProf	$\sqsubseteq$	TeachingStaff
$\perp$	$\equiv$	Professor $\sqcap$ AssistantProf
$\perp$	$\equiv$	IIT_MS_Student $\sqcap$ IITPhdStudent

Table 4

ABox of ACAD ontology

```
IITStudent(tom)
IIT_MS_Student(tom)
hasAdvisor(tom, bob)
IITPhdStudent(sam)
hasAdvisor(sam, alice)
hasAdvisor(sam, roy)
AssistantProf(alice)
```

4. Newly Introduced Terminologies and Definitions

In this section, we introduce the terminologies and definitions by considering ontologies whose expressivity is bound to $\mathcal{SHIQ}$ description logic.

In the current context, 'Description'of an ontology entity refers to its NL definition generated from the ontology.

4.1. Label-sets

To generate descriptions of individuals in an ontology, we associate with each individual a set of constraints it satisfies. We call these sets as *label-sets* in general. A *Label-set* of an individual is called a *node-label-set* and a *label-set* of a pair of individuals is called an *edge-label-set*. The rationale behind generating these label-sets is that, since all the constraints satisfied by an individual are captured at one place, it can easily be looked up for redundancies.

Node-label-set The node-label-set of an individual is the set which contains *all* the class expressions and (existential, universal and cardinality) restrictions satisfied by that individual.

Definition 1 *The node-label-set of an individual x (represented as $\mathcal{L}_\mathcal{O}(x)$) is defined as:*

$$\mathcal{L}_\mathcal{O}(x) = \{c_i \mid \mathcal{O} \models c_i(x)\}$$

where c_i is of the following form:

$$c_i = A \mid \exists R.C \mid \forall R.C \mid \leq nR.C \mid \geq nR.C$$

Here, A is an atomic concept, C is a class expression and R is a role name in ontology $\mathcal{O}$, and m and n are positive integers. C is of the following form:

$$C = A \mid C_1 \sqcap C_2 \mid C_1 \sqcup C_2 \mid \exists R.C_1 \mid \forall R.C_1 \mid \leq nR.C_1 \mid \geq nR.C_1,$$

where C_1 and C_2 are also class expressions.

Note that, in the above definition, the first-level expressions (the $c_i s$) are free from disjunctions. If an individual satisfies a disjunctive clause (a set of independent expressions combined using disjunctions), satisfiabilities of each of these independent expressions are checked and include those expressions that are satisfiable as conjunctions in the label-set. Clearly, the conjunction of all the elements in the label-set of an individual will be entailed by the ontology. That is, $\mathcal{O} \models \big(\sqcap_{i=1}^{n} c_i\big)(x)$

An example of the node-label-set of the individual x = tom from ACAD ontology is: $\mathcal{L}_\mathcal{O}(x)$ = { Student, IITStudent, IIT_MS_Student, $\exists$enrolledIn.IITProgramme, ≤ 1 hasAdvisor. TeachingStaff, $\forall$hasAdvisor.TeachingStaff, $\exists$hasAdvisor.Professor }

Edge-label-set The label-set of a pair of individuals (x, y) is the set that contains *all* the property relationships (role names) from the first individual to the second individual. It is represented as $\mathcal{L}_\mathcal{O}(x, y)$.

Definition 2 *$\mathcal{L}_\mathcal{O}(x, y)$ is formally defined as (where N_R is the set of all atomic roles in ontology $\mathcal{O}$): $\mathcal{L}_\mathcal{O}(x, y) = \{R \mid R \in N_R \wedge \mathcal{O} \models R(x, y)\}$.*

From ACAD ontology, the edge-label-set of the pair (tom, bob) can be written as: $\mathcal{L}_\mathcal{O}$(tom, bob) = { hasAdvisor }.

Although various approaches can be considered for generating label-sets, the practical method that we have adopted for generating the label-sets is explained in the next subsection.

4.2. Label-set generation technique

Node-label-set generation. The naive method to find the node-label-set of an individual is by doing satisfiability check for all combinations of roles, concepts and restrictions types; and include them if they are true. Since, this is not a practically adoptable method for large ontologies, we generate the label-set of an individual x from an ontology $\mathcal{O}$ as follows.

Firstly, we create the corresponding *inferred ontology* $\mathcal{O}'$(using a reasoner). Currently we consider only those consistent ontologies for verbalization. From $\mathcal{O}'$, we find all the concept names and (existential, universal and cardinality) restrictions satisfied by the individual as follows:

Step 1: All the concept names which are satisfied by x are obtained by a simple SPARQL query. We can call it as the *seed* label-set. For example, the set of concept names, which we obtained from $\mathcal{O}'$, corresponding to the individual `tom` is { `Student`, `IITStudent`, `IIT_MS_Student` }.

Step 2: In order to get the restrictions satisfied by x, we access the class definitions and class subsumption axioms corresponding to the concepts which are obtained in the first step, and then consider the existential, universal and cardinality restrictions on the right hand side of those axioms to enrich the label-set.

The right hand side of the axioms in their conjunctive normal form (CNF) is used for enriching the label-set. That is, the R.H.S. will be of the form: $c_1 \sqcap c_2 \sqcap (c_3 \sqcup c_4 \sqcup c_5 \sqcup ... \sqcup c_k) \sqcap c_{k+1} \sqcap ... \sqcap c_{k+n}$. Those clauses in the CNF which do not contain any disjunction, for examples as in c_1, c_2 etc. are directly included in the label-set. If a clause contains disjunction of expressions (denoted as D-Clause), such as $c_3 \sqcup c_4 \sqcup c_5 \sqcup ... \sqcup c_k$ above, then it is handled in parts, as shown in Algorithm 1.

Algorithm 1. Handling disjunctions of expression

procedure LABEL-SET-GEN(x, D-Clause)
  for each expression *exp* in D-Clause **do**
    if *exp* is of the form $\exists R.C$ **then**
      if $\mathcal{O} \models \exists R.C(x)$ **then**
        $\mathcal{L}_{\mathcal{O}}(x) \leftarrow \mathcal{L}_{\mathcal{O}}(x) \cup \{\exists R.C\}$
      end if
    else if *exp* is of the form $\forall R.C$ **then**
      if $\mathcal{O} \models \forall R.C(x)$ **then**
        $\mathcal{L}_{\mathcal{O}}(x) \leftarrow \mathcal{L}_{\mathcal{O}}(x) \cup \{\forall R.C\}$
      end if
    else if *exp* is of the form $\leq nR.C$ **then**
      if $\mathcal{O} \models \leq nR.C(x)$ **then**
        $\mathcal{L}_{\mathcal{O}}(x) \leftarrow \mathcal{L}_{\mathcal{O}}(x) \cup \{\leq nR.C\}$
      end if
    else if *exp* is of the form $\geq nR.C$ **then**
      if $\mathcal{O} \models \geq nR.C(x)$ **then**
        $\mathcal{L}_{\mathcal{O}}(x) \leftarrow \mathcal{L}_{\mathcal{O}}(x) \cup \{\geq nR.C\}$
      end if
    end if
  end for
end procedure

Continuing with our example, enrichment of the label-set of `tom` is done by obtaining existential, universal and cardinality restrictions associated with each of the concept names in the seed label-set. That is, the restrictions $\exists$`enrolledIn.IITProgramme`, ≤ 1`hasAdvisor.TeachingStaff`, $\forall$`hasAdvisor.TeachingStaff`, and $\exists$`hasAdvisor.Professor` associated with concept names are included in $\mathcal{L}_{\mathcal{O}}$(`tom`).

It should be noted that, using this approach, we are generating only those necessary restrictions which can entail the other satisfying combinations as per our label-set definition. For the same reason, we may need to rely on rule-based reasoning (explained later) to generate other restrictions which are of our interest.

Edge-label-set Generation The edge-label-set of a pair of individuals (x, y) can be easily generated from $\mathcal{O}'$ using a simple SPARQL query.

5. Proposed Method for Generating Descriptions

Once we get the label-sets of all the individuals (node-label-sets) in a given ontology, we can generate descriptions of individuals and concepts using the following approaches.

5.1. Description of individuals

Node-label-sets of each individuals are considered for generating their descriptions. Label-sets of all the individuals from ACAD ontology is given in Table 5. For example, by looking at the node-label-set of `tom`, we will get the set of all restrictions (logical expressions) that are satisfied by the individual. Considering these restrictions together, we can frame a meaningful definition for `tom` as: *"Tom is a student who is enrolled in an IIT Programme, has one professor as advisor, and all his advisors are teaching staffs."* Clearly, not all logical expressions (labels) in the label-set are

Table 5

Node-Label-set of individuals in ACAD ontology (intentionally omitted $\top$ class from the label-sets)

$\mathcal{L}_{\mathcal{O}}$(tom)	= { Student, IITStudent, IIT_MS_Student, ∃enrolledIn.IITProgramme, ≤ 1hasAdvisor.TeachingStaff, ∀hasAdvisor.TeachingStaff, ∃hasAdvisor.Professor }
$\mathcal{L}_{\mathcal{O}}$(sam)	= { Student, IITStudent, IITPhdStudent, ∃isEnrolledIn.IITProgramme, ≥ 2hasAdvisor.TeachingStaff, ≤ 1hasAdvisor.Professor, ∀hasAdvisor.TeachingStaff, ∃hasAdvisor.Professor }
$\mathcal{L}_{\mathcal{O}}$(bob)	= { Professor, TeachingStaff }
$\mathcal{L}_{\mathcal{O}}$(alice)	= { AssistantProf, TeachingStaff }
$\mathcal{L}_{\mathcal{O}}$(roy)	= { Professor, TeachingStaff }

necessary to generate such a description. That is, those labels that can induce redundancy in the description can be ignored or combined with other restrictions.

As noted earlier, some of the labels (mainly role restrictions) in the label-set if verbalized directly may generate confusing descriptions, and hence they should be reduced or combined with other restrictions to get a more refined restriction. For example, if left unrefined, the restrictions ∀hasAdvisor.TeachingStaff and ∀hasAdvisor.$\top$ may give rise to the description: *"all advisors are some one and all advisors are teaching staffs"*, which confuses a human reader.

Given a label-set, to generate a refined description, we have to perform two tasks. The first task is to identify the labels that induce redundancy from the label-set. We call this task as *redundant label selection*. The second task is to perform *inferencing* using the selected labels so that they can be combined with the non-redundant labels, to form a refined content.

The naive method to perform the aforementioned tasks is by considering all combinations of labels and see whether they can be reduced or not. This is indeed an exhaustive process since the total number of steps to be taken for completing the refinement depends on the combination which we select at each step. To overcome this, the redundant label selection is carried out by considering labels of specific restriction types in a pre-defined order. For example, all the existential role restrictions are considered prior to universal role restrictions. Such a systematic process along with an ordered list of inference rules that always generate stricter (more specific) forms of a given set of restriction, will ensure a faster refinement of the label-sets. Due to the aforementioned property of the rules, we call them as *refinement-rules*. Since we do this refinement of labels at the logical-level by considering their semantics, we call the two tasks collectively as *semantic-refinement of label-sets*. The refined form of the label-set is called the *semantically-refined label-set*.

The semantic-refinement is not only done to remove redundant labels in a label-set but also to avoid ambiguous verbalization of interim logical expressions. For example, ∀hasAdvisor.Professor is a label which can appear in the label-set of an individual of IITStudent due to the axiom: IITStudent $\sqsubseteq$ ∀hasAdvisor.Professor. Linguistically, this label (along with the axiom) can be interpreted in two ways: either as *All advisors of IIT students are Professors* or, by considering logical equivalent of the statement, it can be interpreted as *Either all advisors of IIT students are Professors or* (vacuously-true case) *they do not have an advisor.* Clearly, including the latter description in the verbalization may confuse a reader. This is especially the case when he could infer from other axioms that the vacuously-true case would not arise.

For identifying the cases where combinations of conditions involving qualifiers and/or number restrictions occur and to succinctly represent them, we introduce the following new constructors.

- Non-vacuous role restriction: $\Im R.C$
  $\Im R.C^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} | \exists y.\langle x, y\rangle \in R^{\mathcal{I}} \wedge y \in C^{\mathcal{I}} \wedge \forall z.\langle x, z\rangle \in R^{\mathcal{I}} \implies z \in C^{\mathcal{I}}\}$
- Exactly-one role restriction: $\exists_{=1} R.C$
  $\exists_{=1} R.C^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} | (\exists y_1.\langle x, y_1\rangle \in R^{\mathcal{I}} \wedge y_1 \in C^{\mathcal{I}} \wedge \exists y_2.\langle x, y_2\rangle \in R^{\mathcal{I}} \wedge y_2 \in C^{\mathcal{I}}) \implies y_1 = y_2\}$
- Exactly-n role restriction: $\exists_{=n} R.C$, general case of exactly-one role restriction.

In our semantic refinement process, like any rule-based approach, the order in which the inferencing rules are applied is also important as the applicability of one rule may depend on the other. We observed that there is a notion of *strictness* associated with role restrictions which can be effectively utilized for ordering the rules, such that the redundant label selection and the application of the rules can be done simultaneously. The notion of strictness can be looked at as: if a role restriction R_1 is implied by another role restriction R_2 (i.e., $R_2 \implies R_1$), then R_1 can be said as a stricter version of R_2. For instance, $\Im R.U$ can be said as the stricter form of $\exists R.U$ and $\forall R.U$. Similarly, $\exists_{=n} R.U$ is a sticker form of $\leq nR.U$ and $\geq nR.U$. Since we intend to find sticker forms of role-restrictions, the obvious way is to apply rules corresponding to less stricter restriction types prior to those of stricter restriction types.

In the forthcoming sub-section, we introduce our rule-based refinement algorithm to accomplish complete reduction where we do all the possible reduction of less stricter restrictions prior to reducing stricter ones. Completeness of the refined form of label-set is guaranteed by the construction of the algorithm.

In what follows, we discuss how semantic-refinement of label-sets can be achieved.

5.1.1. Semantic-refinement of label-sets

We propose seven sets of rules for refining a label-set. Each of these rule sets contain carefully chosen rules which are repeatedly applied on the selected restrictions in the label-set until no more refinement is possible. On moving from one rule set to another, those labels which have been reduced already would be *provisionally* removed from the label-set. More details about the algorithm is given in the next sub-section.

The details of the first five sets of rules are given in Table 6. Each of the rule sets are named based on the type of restriction they handle. For example, the first rule set is called *Concept Refinement rule* since it refines the atomic concepts in the label-set. More details about the refinement rules are given below.

Concept Refinement Rule (Rule 1a). To apply this rule, the redundant label selection process is to consider all the concept name symbols that are present in the label-sets whose definitions (i.e., the set of restrictions which defines the concept) are included in the label-set. If the defining restrictions of a concept are present in the label-set, we can apply the rule and the corresponding concept names can be removed, since they are redundant content.

Superclass Refinement Rule (Rule 2a). Consider the individuals given in Table 5, we can see that their label-sets contain all the concept names which they belong to. Some of the concepts in these label-sets are hierarchically related (in class - super-class relationship) in the ontology, resulting in redundant labels. For example, consider the label-set $\mathcal{L}_{\mathcal{O}}(\texttt{tom})$, it contains the concepts `IIT_MS_Student` and `IITStudent`. Since it can be inferred from the concept `IIT_MS_Student` that `tom` is also a `IITStudent`, we can say that `IITStudent` is a redundant information (label) in the label-set. We remove such redundant labels by using most-specific concept notion, for that we first identify all the possible concept chains that are present in the label-set. It should be noted that an individual may present in 2 or more such subsumption concept chains. By applying the refinement rule to each chain we maintain only the most-specific concepts.

(Note that, this refinement rule is applied only after the applications of the concept refinement rule – some specialized concepts may get removed while applying the rules in the first rule set, therefore, it does not always mean that a refined label-set contains only the specialized concept names in the ontology)

The presence of redundant concept names in a node-label-set is mainly because, we do a classification on the ontology prior to the label-set generation.

The upcoming rule sets are meant for reducing the various role restrictions allowed in a $\mathcal{SHIQ}$ ontology.

Existential Role Refinement rule (Rule 3a). We can select two labels of the form: $\exists R.U$ and $\exists S.V$, from the label-set, as candidates for applying this rule, if $U \sqsubseteq V$ and $R \sqsubseteq S$, in the ontology. According to the existential role refinement rule, candidate labels are semantically equivalent to stating only a single restriction of the form $\exists R.U$ (which we call as the *refined form* of the labels). In general, all the rules that we cover in this paper are defined such that given a refined form and the condition which have been used for refinement, the non-refined forms of the restriction(s) can be traced back. This means that, the refinement is done without affecting the semantics/meaning of the restrictions.

Formally, the correctness of the existential role refinement rule can be proven as follows:

Proof of Rule 3a. Given an ontology $\mathcal{O}$ with R and S as its roles, and U and V are two of its concepts, and $\mathcal{O} \models U \sqsubseteq V, R \sqsubseteq S$, then $\exists R.U \sqcap \exists S.V \equiv \exists R.U$. To prove this, let us consider an individual $x \in \exists R.U \sqcap \exists S.V$, clear it implies $x \in \exists R.U$. Therefore $\exists R.U \sqcap \exists S.V \sqsubseteq \exists R.U$. Now, if $x \in \exists R.U$, it implies that there exist an arbitrary a, such that $(x, a) \in R, a \in U$. Since $U \sqsubseteq V$, we can say that $a \in V$. It implies,

Table 6

Details of rule sets 1-5.

Rule No.	Restriction 1	Restriction 2	Condition	Refined form
Concept Refinement rule				
1a	Concept names, whose (equality) definitions are already included in the label-set, can be removed.			
Superclass Refinement rule				
2a	U	V	$U \sqsubseteq V$	U
Existential Role Refinement rule				
3a	$\exists R.U$	$\exists S.V$	$U \sqsubseteq V$ & $R \sqsubseteq S$	$\exists R.U$
Universal Role Refinement rules				
4a	$\forall R.U$	$\forall S.V$	$U \sqsubseteq V$ & $S \sqsubseteq R$	$\forall R.U, \forall S.U$
4b	$\forall R.U$	$\forall R.V$	$V \sqsubseteq U$	$\forall R.V$
III & IV Combination rules				
5a	$\exists R.U$	$\forall R.U$		$\Im R.U$
5b	$\forall R.U$	$\exists S.V$	$U \sqsubseteq V$ & $S \sqsubseteq R$	$\Im R.U, \Im S.U$
5c	$\forall R.U$	$\exists S.V$	$V \sqsubseteq U$ & $S \sqsubseteq R$	$\Im R.U, \exists S.V$

Table 7

Details of rule sets 6 and 7.

Rule No.	Restriction 1	Restriction 2	Condition	Refined form
Qualified Number Restriction Refinement rules				
6a	$\geq nR.U$	$\geq mS.V$	$U \sqsubseteq V$ & $R \sqsubseteq S$ & $n \geq m$	$\geq nR.U$
6b	$\exists R.U$	$\geq nS.V$	$V \sqsubseteq U$ & $S \sqsubseteq R$ & $n \geq 1$	$\geq nS.V$
6c	$\exists R.U$	$\leq nR.V$	$U \sqsubseteq V$ & $n = 1$	$\exists_{=1} R.U, \exists_{=1} R.V$
6d	$\geq nR.U$	$\leq nS.V$	$R \sqsubseteq S$ & $U \sqsubseteq V$	$\exists_{=n} R.U, \exists_{=n} S.V$
Exactly-n Role Refinement rules				
7a	$\exists R.U$	$\exists_{=1} S.V$	$U \sqsubseteq V$ & $R \sqsubseteq S$	$\exists_{=1} R.U, \exists_{=1} S.V$
7b	$\Im R.U$	$\exists_{=1} S.V$	$U \sqsubseteq V$ & $R \sqsubseteq S$	$\exists_{=1} R.U, \exists_{=1} S.V, \Im R.U$
7c	$\geq mR.V$	$\exists_{=n} R.U$	$U \sqsubseteq V$ & $m \geq n$	$\exists_{=n} R.U, \geq (m-n)R.(V \sqcap \neg U)$

$x \in \exists S.V$. Similarly, since $R \sqsubseteq S$, $(x,a) \in R \implies (x,a) \in S$ Therefore, $\exists R.U \sqsubseteq \exists R.U \sqcap \exists S.V$.

Universal Role Refinement rules (Rules 4a and 4b). This rule set contains two rules which help in refining universal role restrictions. If a label-set contains two role restrictions of the form: $\forall R.U$ and $\forall S.V$, universal role refinement rules can be applied if they satisfy the conditions of the rule. For example, if the label-set contains ∀`hasAdvisor.Professor` and ∀`hasAdvisor. TeachingStaff`, and if `Professor` ⊑ `TeachingStaff`, we can refine those restrictions to ∀`hasAdvisor. Professor`. The correctness of the two rules can be easily be proven as follows.

Proof of Rule 4a. Given an ontology $\mathcal{O}$ which entails $U \sqsubseteq V$ and $S \sqsubseteq R$ (where R and S are roles, and U and V are concepts), then $\forall R.U \sqcap \forall S.V \equiv \forall R.U \sqcap \forall S.U$. Proving $\forall R.U \sqcap \forall S.U \sqsubseteq \forall R.U \sqcap \forall S.V$ is trivial since $\forall S.U \sqsubseteq \forall S.V$ (given, $U \sqsubseteq V$). Now, let $x \in \forall R.U \sqcap \forall S.V$, suppose $(x,a) \in S$ where a is an arbitrary individual. Since $S \sqsubseteq R$, $(x,a) \in R$. It implies $a \in U$ (since $x \in \forall R.U$). Therefore, we get $x \in \forall S.U$. Hence, $\forall R.U \sqcap \forall S.V \sqsubseteq \forall R.U \sqcap \forall S.U$.

Proof of Rule 4b. Given an ontology $\mathcal{O}$ which entails $V \sqsubseteq U$ (where R is a role and, U and V are concepts), then $\forall R.U \sqcap \forall R.V \equiv \forall R.V$. Proving $\forall R.U \sqcap \forall R.V \equiv \forall R.V$ is trivial, since the L.H.S. can be written as $\forall R.(U \sqcap V)$, and it is equivalent to $\forall R.V$, since $V \sqsubseteq U$.

Further in this section, we refrain from giving the proof of correctness of the rules in the succeeding rule sets. An appendix is provided at the end of the paper with all the required proofs.

III & IV Combination rules (Rules 5a, 5b and 5c). For applying the rules in this rule set, the redundant label selection process selects combinations of existential and universal role restriction from the label-set. The rules help in refining such combinations to a reduced form.

The details of the next set of rule sets are given in Table 7.

Qualified Number Restriction Refinement rules. In this set there are four rules. Here we mainly try to refine qualified number restriction restrictions (of the form $\leq nR.U$ or $\geq mS.V$) to stricter version of the same form or to a exactly-n restrictions.

Exactly-n Role Restriction rules. In this rule set, we reduce the exactly-n role restrictions which are generated using the preceding rule-sets. The rule set is named so because, this is the only rule set where we try to reduce exactly-n role restrictions.

5.1.2. Algorithm for semantic-refinement

As we mentioned before, semantic-refinement helps in refining restrictions in a label-set to their stricter forms by combining them using a set of rules. The rules are applied sequentially from rule-set 1 to 7. While applying these rules, some of the reduced restrictions are removed provisionally to avoid exhaustive operations in the forthcoming iterations. In our algorithm, we mark such restrictions as PRs (Provisionally Reduced ones), so that at a later stage we can remove them permanently from the label-set.

Algorithm-2 describes the steps that has to be followed for applying the rules. This algorithm works by taking pairs of restrictions from the label-set, and looking for the applicability of the rules. If a rule is applicable, the restrictions will be checked for the following set of conditions, to decide whether to resume the refinement or not. These conditions are followed mainly to ensure quick refinement.

Condition-1. No need to further reduce two provisionally reduced (PR) restrictions.

The rule-sets are designed in such a way that if a particular combination of restriction types is reduced by a rule in one rule-set, the same combination will not occur in the succeeding rule-sets. Condition-1 is based

Algorithm 2. Semantic-refinement of label-sets

1: **procedure** SEMANTIC_REFINEMENT($\mathcal{L}_\mathcal{O}$(x))
2:   Mark all $u \in \mathcal{L}_\mathcal{O}(x)$ as not PRs
3:   Apply *Concept Refinement rule* and remove appropriate concept names from $\mathcal{L}_\mathcal{O}$(x)
4:   $R \leftarrow$ Rule-sets 2-7 ▷ list of pre-defined rules
5:   **for each** rule-set $rs \in R$ **do**
6:     Let $M, REF \leftarrow \phi$
7:     **for each** $(u, v) \in \mathcal{L}_\mathcal{O}(x)\times\ \mathcal{L}_\mathcal{O}(x)$ AND $u \neq v$ **do**
8:       **if** (NOT(MARKED_AS_PR(u)) AND NOT(MARKED_AS_PR(v))) **then**
9:         **for each** $(r \in rs)$ **do**
10:           **if** r is applicable on (u, v) **then**
11:             $M \leftarrow$ APPLY_RULE(r, u, v)
12:             $\mathcal{L}_\mathcal{O}(x) \leftarrow \mathcal{L}_\mathcal{O}(x) \cup M$
13:             $REF \leftarrow REF\ \cup$ {u,v}
14:             **if** $u \in M$ **then**
15:               $REF \leftarrow REF \backslash \{u\}$
16:             **end if**
17:             **if** $v \in M$ **then**
18:               $REF \leftarrow REF \backslash \{v\}$
19:             **end if**
20:           **end if**
21:         **end for**
22:       **end if**
23:     **end for**
24:     MARK_AS_PR(REF)
25:     $\mathcal{L}_\mathcal{O}(x) \leftarrow \mathcal{L}_\mathcal{O}(x)\ \cup\ REF$
26:     **for each** $u \in \mathcal{L}_\mathcal{O}(x)$ **do**
27:       **if** the restn. type of u is not used in the successive rule-sets AND MARKED_AS_PR(u) **then**
28:         $\mathcal{L}_\mathcal{O}(x) \leftarrow \mathcal{L}_\mathcal{O}(x) \backslash \{u\}$
29:       **end if**
30:     **end for**
31:   **end for**
32: **end procedure**

on this property of the rule-set and the construction of Algorithm 2. In the algorithm, at first a rule-set (in order) would be selected and all the restrictions are selected in pairs to see if a refinement based on the rules in the selected rule-set is possible or not. Assume if the condition (in line 8) is not present, then, all the pairs of restrictions that are less stricter forms based on the current restriction types in the rule-set would need to be considers for checking the applicability of the rules which is however not necessary.

Condition-2. If a rule combines two restrictions ($R1$ and $R2$) and generates either $R1$ or $R2$, then that $R1$ or $R2$ should not be marked as a PR.

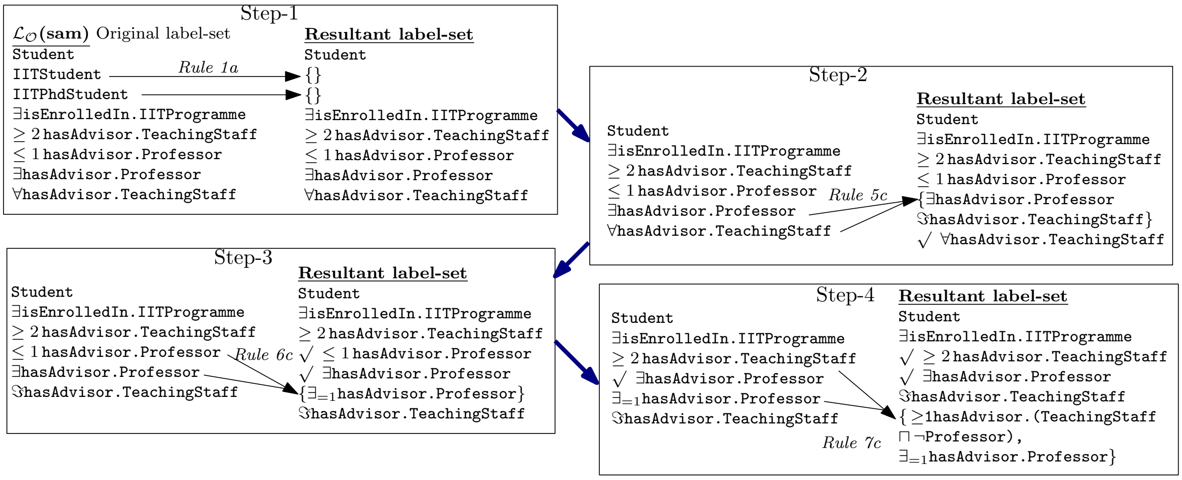

Fig. 1. Steps involved in the semantic-refinement of $\mathcal{L}_{\mathcal{O}}$(sam). Arrows represent the application of rules.

This is a special case for the MARK_AS_PR function that we introduce in Algorithm 2. This condition should be guaranteed while marking an restriction as PR because of the following reason. In the rules such as 4a, 5c, 6a etc., one of their antecedent term gets repeated in the consequent part of the rule to ensure reverse implication (i.e., for preserving semantics). On applying such rules, if the regenerated terms are marked as PRs, they may get permanently removed during the course of the algorithm, which is not acceptable.

Condition-3. If the restrictions of a particular form are *not* used in successive rule-sets, the PR restrictions of such forms can be removed at an early stage.

This condition can be checked at the MARK_AS_PR function. This condition can be considered because, across two rule-sets (such as 5 and 6) same restrictions types are appearing in their antecedents. As the condition says if a restrictions are not used later, either they can be removed after the applications of rules in all the rule-sets or they can be removed at specific points where it can be determined that they will not be used by any rules from then on. The latter would be reduce more inner iterations.

For illustration, let us consider the node-label-set of the individual `sam`. Fig 1 shows the refinement steps and the rules in the rule sets which are used for the refinement. $\mathcal{L}_{\mathcal{O}}$(`sam`) is represented vertically. In the figure, the arrows represent the application of rules. Rule numbers are represented in italics. A refinement of two restrictions may sometimes result in more than one restrictions, to represent them, the arrows are followed by brace brackets ({...}) to show the resultant restrictions.

Initially, the algorithm marks all the labels in the label-set as not PRs. Then the algorithm looks for the applicability of the rule 1a (concept refinement rule). In the figure, $\mathcal{L}_{\mathcal{O}}$(`sam`) contains the labels `IITStudent` and `IITPhdStudent` whose definitions (in the form of restrictions) are already present in the label-set. Therefore, the Rule 1a is applied on those labels and can remove them from the label-set. In the algorithm, lines 5-31 take the rest of the rule-set one at a time, and look for possible application of rules on pairs of restrictions in the label-set. In our example label-set, since no rules in the rule-sets 2,3, and 4 are applicable, we move to the next applicable rule set, rule set number 5. Now, the algorithm applies the rule 5c on two of the restrictions as shown in the figure and refine them to the two restrictions given in the brackets. Application of a rule will be done only if the restrictions in the pair are not marked as PR (checked using the function MARKED_AS_PR(.)). The *if* condition in the line-8 of the algorithm will take care of this. After the application of a rule (using the function APPLY_RULE(.)), the details of the reduced restrictions will be stored in the set variable REF. Based on the condition-2, appropriate changes have to be done on the contents of REF (lines 14-20). Once all the possible rules in a particular rule set are applied, the reduced restrictions will be marked as PRs (lines 24). Once the algorithm considers all pairs of labels and checks them for the applicability of all the rules in the current rule-set, the condition-3 will be checked for possible permanent removal of the PRs. The entire process will be repeated for all the succeeding rule-sets.

Coming back to our example label-set, after the application of Rule 5c, one of the reduced restriction is marked as PR (represented using $\surd$), while the other restriction is not marked as PR due to the condition-2. On changing the rule-set, since no other rules in rule-set 5 are applicable, the one which is marked as PR can be permanently removed since the condition-3 is satisfied. In the forthcoming iterations of the for loop (line 5), rules in the rule-set 6 and 7 are applied in similar fashion. In the last iteration, we will get the most refined set of labels, along with a set of restrictions which are marked as PRs. The restrictions which are marked as PRs are removed to get the refined label-set.

Illustration of the usefulness of the approach. The usefulness of semantic refinement can be illustrated by looking at the sentences that can be generated from the node-label-set before and after refinement. Considering the original node-label-set, sam can be defined as "*A student, an IIT student, an IIT PhD student, who is enrolled in an IIT programme, has more than two advisors who are teaching staffs, has less than one and at least one advisor who is a professor, and all advisors are teaching staff*". By making use of the refined node-label-set, we can generate a smaller and easily-understandable definition: "*A student who is enrolled in an IIT programme, has exactly one advisor who is a professor and has at least one more advisor who is a teaching staff but not a professor*". More examples and evaluation results to support the usefulness of this approach are presented in Section 7.

5.2. *Description of Concepts*

A concept can be defined in a similar fashion as that of an individual using label-sets. To generate the description of a concept, we introduce a new individual as its member. It is important that the new individual should be assigned as the member of only the concept whose definition has to be found. Now, label-set corresponding to this newly introduced individual is utilized to generated the concept's definition. The rationale behind introducing a new individual is that, in order to find the definition of a concept (say Concept A), we only need restrictions which are associated with it and its super-classes. Considering an existing individual may result in a case where it may belong to concepts which are sub-classes of the concept A; this results in including the restrictions associated with the specific-classes also in the label-set, which is undesirable. Introducing a new individual will overcome this issue; in addition, the approach will even work smoothly for those concepts that do not have an individual.

Let us look at an illustration of generating definition of IITPhdStudent from ACAD ontology. At first, we introduce the individual ips as a member of IITPhdStudent. Now we will find the label-set of ips.

We get $\mathcal{L}_{\mathcal{O}}$(ips) as {Student, IITStudent, IITPhdStudent, $\exists$isEnrolledIn.IITProgramme, ≥ 2 hasAdvisor.TeachingStaff, ≤ 1 hasAdvisor.Professor, $\forall$hasAdvisor.TeachingStaff, $\exists$hasAdvisor.Professor}

In the next step, we remove the concept name, whose definition has to be found, from the obtained label-set. That is, $\mathcal{L}_{\mathcal{O}}(ips)\setminus${IITPhdStudent}. This new label-set is semantically-refined and verbalized to get the redundant-free description of the concept.

Therefore, IITPhdStudent can be defined as: { Student, $\exists$isEnrolledIn.IITProgramme, $\exists_{=1}$hasAdvisor.Professor, $\Im$hasAdvisor.TeachingStaff, ≥ 1hasAdvisor.(TeachingStaff $\sqcap\neg$Professor)}

Even though this approach works well for those concepts whose (axiomatized) definitions contain only conjunctive clauses, it may generate incomplete descriptions when the definition contains a disjunctive clause. For example, if the definition of the concept IITStudent is of the form IITStudent $\equiv$ $\exists$isEnrolledIn.IITProgramme $\sqcap$(IITPhdStudent $\sqcup$ IIT_MS_Student), the label-set of a newly introduced individual of IITStudent (say, stud) should be {IITPhdStudent $\sqcup$ IIT_MS_Student, $\exists$isEnrolledIn.IITProgramme}. However, our current label-set generation method will not include disjunctive clauses as such in the label-set, instead it will look for the satisfiability of each of the expression in the disjunctive clause (that is, IITPhdStudent(stud) and IIT_MS_Student(stud)), and include them in the label-set, if they are true. But, for stud, they will not be true as we are not explicitly adding any other facts into the ontology other than IITStudent(stud). Therefore, we will get the label-set as {IITStudent, $\exists$isEnrolledIn.IITProgramme} which is an incomplete label-set of the concept. On doing the next steps – removing the concept name itself from the label-set, and doing a semantic-refinement over it – the incompleteness persists. To overcome issue, after semantic refinement step, we will enrich the refined label-set with the previously encountered disjunctive

Table 8
Refined node-label-sets of individuals in ACAD ontology

Individual	Refined-label-set
sam	{ Student, ∃isEnrolledIn.IITProgramme, $\exists_{=1}$hasAdvisor.Professor, ℑhasAdvisor.TeachingStaff, $\geq$ 1hasAdvisor.(TeachingStaff ⊓¬Professor)}
tom	{ Student, ∃isEnrolledIn.IITProgramme, $\exists_{=1}$hasAdvisor.Professor }
bob	{ Professor }
alice	{ AssistantProfessor }
roy	{ Professor }

clause(s). That is, we get the new refined label-set of `stud` as {`IITPhdStudent ⊔ IIT_MS_Student`, `∃isEnrolledIn.IITProgramme`}.

6. Natural Language Descriptions from the Refined Label-sets

In this paper, prime focus is given for the generation of redundancy-free descriptions of ontology entities represented in the form of logical expressions. Appropriate NL sentence generation of these logical forms is yet to be fully explored. However, for the completeness of the paper, we present a simple method which we have adopted to generate NL descriptions of individuals and concepts from their refined label-sets.

Table 9
Constraint-specific templates of the possible restrictions in a redundancy-free description-set.

Restrictn.	Constraint-specific template
$\exists R.C$	<R-verb> at least one <C> as <R-noun>
$\forall R.C$	<R-verb> only <C> as < R-noun >
$\geq nR.C$	<R-verb> at least <n><C> as <R-noun>
$\leq mR.C$	<R-verb> at most <m><C> as <R-noun>
$\Im R.C$	<R-verb> at least one <C> and only <C> as <R-noun>
$\exists_{=n} R.C$	<R-verb> exactly <n><C> as <R-noun>

NL description of an entity is defined as the set of NL fragments which describes the class names and role restrictions it satisfies. An example of a description of `tom` is:

> `tom`: is a student, enrolled in at least one IIT programme, and has exactly one professor as advisor

We consider a template similar to the following regular expression (abbreviated as regex) for generating descriptions of individuals and concepts.

`Individual/concept`: (“is”) ((“a”) ClassName (“,” | “and”)?)$^+$ (RoleRestriction (“,” | “and”)?)$^+$

In the above regex, ClassName specifies the concept names in the label-set. We use the `rdfs:label` role values of the class names as the ClassName. If `rdfs:label` role is not available, the local names of the URIs are used as the ClassName. For RoleRestriction, the role restrictions in the label-set are utilized. The role restrictions are treated in parts. We first tokenize the role names in the constraints. Tokenizing includes word-segmentation and processing of camel-case, underscores, spaces, punctuations etc. Then, we identify and tag the verbs[5] and nouns in the segmented phase — as R-`verb`, R-`noun` respectively — using the Natural Language Tool Kit[6]. We then incorporate these segmented words in a *constraint-specific template*, to form a RoleRestriction. For instance, the restriction $\exists$*hasAdvisor.Professor* is verbalized to “has at least 1 professor as advisor”, using the template: <R-`verb`> at least <n><C> as <R-`noun`> (where C corresponds to the concept present in the restriction). Constraint-specific templates corresponding to the possible restrictions in a label-set are listed in Table-9. In our studies, we have also tried out variants of these constraint-specific templates to further tune the NL output. Since the empirical study (see the next section) is done for a different intention, involving only a carefully chosen participants, we refrain from further enhancing the fluency of the NL texts.

If the C equivalent portion of the restriction is not a concept name (atomic concept), that is, if it a conjunction or disjunction of restrictions, Table 9 will be recursively looked up for possible templates, and the

[5]In the absence of a proper verb, the phrase “related to” is used in its place.
[6]Python NLTK: http://www.nltk.org/

Table 10

Examples of the descriptions of individuals and concepts from PD, HP and GEO ontologies, generated using the proposed and as well as the traditional approaches

Entity type	*Proposed approach*	*Traditional approach*	*Ontology*
Indivl.	*Bird cherry Oat Aphid:* is a biotic-disorder, having at least one pest-insect and all its factors are pest-insects.	*Bird cherry Oat Aphid:* is a disorder, bio-disorder, pest damage and insect damage. It is all the following: has as factor only pest-insect, has as factor only pest, has as factor only organism and has as factor something.	PD
Indivl.	*Black Chaff:* is a plant disease, having at least one bacteria and all its factors are bacterium.	*Black Chaff:* is a disorder, a biotic disorder, a plant disease and a plant bacterioses. It is all the following: has as factor only organism, has as factor only micro-organism and has as factor only bacterium, has as factor at least one thing	PD
Concept	*Mite Damage:* is a pest damage, having at least one mite pest and all its factors are mite pests.	*Mite Damage:* is a disorder, a biotic-disorder and a pest damage. It is all the following: has as factor only organism, has as factor only pest, has as factor only mite pest, has as factor at least one thing.	PD
Indivl.	*Hermione Granger:* is a Hogwarts Student, a muggle, a gryffindor, having exactly one cat as pet.	*Hermione Granger:* is a Hogwarts student, a student, a human, a muggle, a gryffindor. It is all the following: has a pet, has as pet a cat, has as pet only creature, has at least one creature, has at most one creature, as pet.	HP
Concept	*Hogwarts Student:* is a Student, is a Gryffindor or Hufflepuff or Ravenclaw or Slytherin, and having exactly one pet.	*Hogwarts Student:* is a student, a human, is a Gryffindor or Hufflepuff or Ravenclaw or Slytherin. It is all the following: has a pet, has as pet only creatures, has at least one creature, has at most one creature.	HP
Indivl.	*Hedwig:* is an owl, is related to at least one Hogwarts student and only Hogwarts student, as pet.	*Hedwig:* is an owl, a pet, a creature. It is all the following: is pet of only Hogwarts student, is pet of a Hogwarts student.	HP
Concept	*Aggregate of sovereign states:* is not a governmental organization, is aggregate of only sovereign states and is aggregate of at least two sovereign states.	*Aggregate of sovereign states:* is not a governmental organization and not a sovereign state. It is all the following: is aggregate of only governmental organization, is aggregate of at least two governmental organizations, is aggregate of only sovereign states and aggregate of at least two sovereign states.	GEO
Indivl.	*Florida:* is a government organization and a major administrative subdivision, is related to at least one nation as a part, is related to exactly one sovereign state as a member, and is a subordinate authority of at least one sovereign state.	*Florida:* is a major administrative subdivision, an organization, a governmental organization, a subnational entity. It is all the following: is a part of at least one nation, is a subordinate authority of at least one sovereign state, is a member of at least one sovereign state and have at most one member of relationship with sovereign state.	GEO

conjunctions and disjunctions will be replaced with 'and'and 'or' respectively.

When it comes to generating concept definitions, we can expect clauses containing disjunctions (independent expressions combined using disjunctions) in the refined label-set. They are handled in parts by taking each of those independent expressions in the clause separately for NL generation, and, they are then combined using 'or'.

7. Empirical Evaluation

We present two case studies to explore the applicability of the redundancy-free description of individuals and concepts in validating the domain knowledge. Rather than choosing an ontology under development, we study the cases of validating two previously built ontologies.

In the study, domain experts were presented with two representations of the same knowledge: one is by direct verbalization of the label-sets and the other is by verbalizing them after finding the corresponding refined label-sets. Direct verbalization of a label-set generates texts (or descriptions) which are similar to those texts which are produced by an existing ontology verbalizer — we call this method as *traditional approach*, and the other as the *proposed approach*. Examples for the description texts that are generated using the proposed approach and traditional approach, from the Plant Disease (PD) ontology[7], HarryPotter (HP) ontology[8] and Geographical Entity[9] (GEO) ontologies are given in Table 10. One can clearly see that those descriptions which are generated using the proposed approach are compact, precise and easy-to-understand when compared to those which are generated using the traditional approach.

Scope of the study. We have done the empirical study mainly for two reasons. Firstly, for finding whether the process of semantic-refinement is helpful in generating useful texts for describing the ontology. For this purpose, the experts were asked to rate their degree of understanding of the knowledge in the scale: (1) poor; (2) medium; (3) Good.

Secondly, to measure the usefulness of the generated sentences (i.e., the descriptions of individuals and concepts) in validating the domain knowledge, domain experts were told to choose from the options: (1) Valid (2) Invalid (3) Don't know (4) Cannot be determined. Significance of these options is that, if a participant is choosing the 4th option, it is likely that she finds it difficult to reach a conclusion on the validity of the sentence presented. In addition, feedbacks are collected from the experts to get suggestions on improving the system.

Dataset used. We used two ontologies for generating descriptions. We have chosen the ontologies in such a way that all the rules in the rule sets are applied at least once during the refinement process. The first ontology is Plant-Disease ontology (PD ontology) developed by International Center of Agricultural Research in the Dry Areas (ICARDA), and the second one is a synthetic ontology, Data structures and Algorithms (DSA) ontology, developed by ORG group[10] at IIT Madras[11]. These ontologies are available at our research group's website[12]. The current version of PD ontology has 546 individuals, 105 concepts and 15 object properties. The DSA ontology has 333 individuals, 53 concepts, 19 object properties and 11 datatype properties.

Experimental setup. For each of the individuals and concepts in the two ontologies we have generated corresponding NL descriptions from their node label-sets as well as from their refined label-sets, using an implemented prototype of the system. Since manual evaluation of all the generated descriptions is difficult, a selected number of descriptions were utilized for the study. (Since the definitions are based on TBox axioms alone (but not their ABox relationships), it is possible to have groups of individuals who are having same label-sets. On verbalizing such label-sets, those individuals would have same definitions. We have ignored such cases so that we will have a heterogeneous set of definitions for our experiments.) The set of descriptions of individuals for the study were selected by grouping the entire descriptions based on their label-sets and randomly choosing one individual's description from each group. The set of descriptions of concepts were selected from those set of descriptions (generated from refined label-sets) which are highly different from their counterparts that are generated from their non-refined label-sets.

From PD ontology, 31 descriptions of individuals and 10 descriptions of concepts have been considered for evaluation. Similarly, for DSA ontology, 14 descriptions of individuals and 17 descriptions of concepts were chosen for evaluation. Then, experts of the two domains were asked to review the verbalized descriptions. To avoid biasing, the reviewers were not aware which definitions were generated from which approach and the definitions are randomly presented using a google form. In addition, we record the confidence score of each reviewer for a given question such that in the case of conflict we make a decision based on their scores. Majority ratings of the sentences were considered for finding the statistics. It would be very interesting to calculate inter-rater agreement, however, we have not done that for this experiment.

Expert selection. Seven experts of plant disease areas and fourteen experts of data structures and algorithms were involved in the study. The seven experts of PD domain have either a masters degree or a doctorate degree in the plant disease or agriculture related

[7]http://wiki.plantontology.org/index.php/Plant_Disease_Ontology
[8]https://sites.google.com/site/ontoworks/ontologies
[9]https://bitbucket.org/uamsdbmi/geographical-entity-ontology/src (last accessed: 27/11/2015)
[10]https://sites.google.com/site/ontoworks/home
[11]https://www.iitm.ac.in/
[12]https://sites.google.com/site/ontoworks/ontologies

areas. The fourteen experts of DSA domain have successfully completed the advanced data structures and algorithms course offered at IIT Madras.

7.1. *Results and Discussions*

Fig 2-5 show the summary of the ratings given by the domain experts. Based on this data, we answer the following two questions.

7.1.1. *How does the semantic-refinement help in improving the understandability of the verbalized knowledge?*

The degree of understanding of each of these descriptions to the domain experts can be identified by looking at the ratings (i.e., poor, medium or good) which they had chosen during the empirical study. If there exists an ambiguity in the description (due to its verbatim fidelity to OWL statements), they are expected to choose poor or medium as the level of understanding. To confine the reasons for ambiguity to the fidelity to OWL constructs alone, possible (manual) grammatical error corrections have been done on the generated text — as we were not using any sophisticated NL generation techniques. Grammatical errors such as subject-verb agreement errors, verb tense errors, verb form errors, singular/plural noun ending errors and sentence structure errors were corrected.

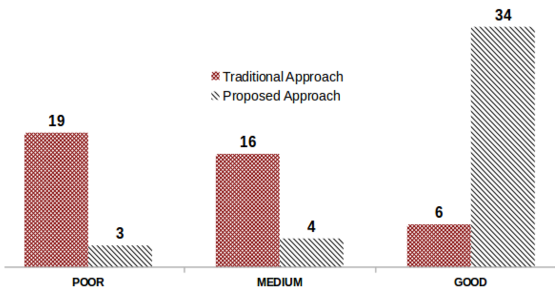

Fig. 2. Y-axis shows the count of descriptions of a particular rating which are generated using our *proposed approach* and the *traditional* approach from the **PD** ontology

Fig 2 shows the overall responses which we received from the seven domain experts for the descriptions of PD ontology. We call it as the overall response because, ratings are calculated by looking at the majority responses; that is, only if a description is rated as 'good' by at least 4 participants, it will be considered as a good description; similar is the case with poor and medium ratings. The dotted-bars represent the count of the descriptions of a particular rating which are generated using the proposed approach and the stripped-bars denote the count of those which are generated using the traditional approach. Similarly, Fig 3 shows the summary of the responses received for DSA ontology. For PD ontology, out of 41 descriptions which are generated using the proposed approach, 34 were rated as 'good', whereas for those which are generated using the traditional approach, only 6 out of 41 texts were rated as 'good'. For DSA ontology, 24 out of 31 descriptions generated by proposed approach are 'good', only 11 descriptions that are generated using the traditional approach were rated as 'good'. These results highlight the significance of the semantic-refinement process in domain knowledge understanding.

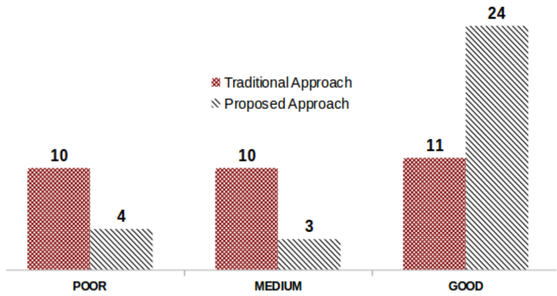

Fig. 3. Y-axis shows the count of descriptions of a particular rating which are generated using our *proposed approach* and the *traditional* approach from the **DSA** ontology

Statistical analysis. We observed that 34 out of 41 and 24 out of 31 descriptions from two samples are found to have good understandability. Therefore, we can hypothesize that 80 percent of the descriptions (in the population) generated after semantic refinement will have good understandability. This claim needs to be tested.

The data collected for the chi-square test is summarized in Table 11. As observed data, we considered 72 samples arbitrary taken from the descriptions of PD and DSA ontologies, and the degree of freedom is one (since we consider only two categories: 1. *Good* sentence and 2. Medium or Poor – i.e., *Not-Good* sentence). In Table 11, the values 57.6 and 14.4 denote the expected count of Good and Not-Good sentences – i.e., 80% and 20% of the sample size – respectively.

The null hypothesis H_o is same as what we hypothesized above, and the alternate hypothesis, H_a, is that semantic refinement does not help in improving the understandability as we claimed. The chi-square score is obtained as 0.014 and the P-value is 0.90618562.

The null hypothesis is not rejected since the P-value is greater than the significance level (0.05).

Table 11
Summary of observed and expected data for statistical analysis – finding understandability

Category	Observed	Expected
Good	58	57.6 (80%)
Medium + Poor	14	14.4 (20%)

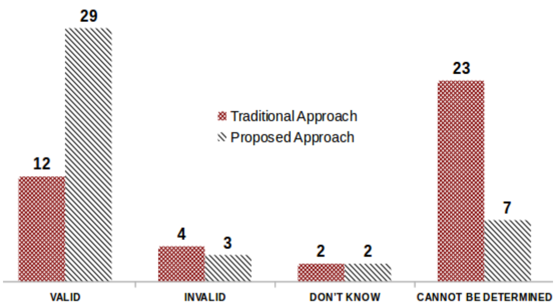

Fig. 4. Statistics (based on the majority responses) to determine the usefulness of the generated descriptions in validating the **PD** ontology

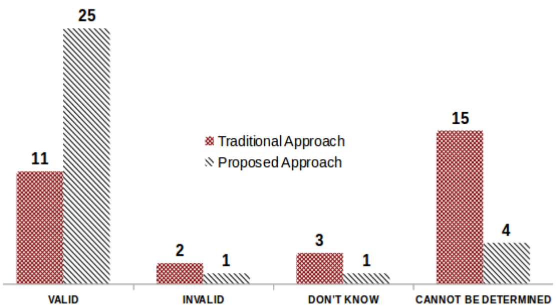

Fig. 5. Statistics (based on the majority responses) to determine the usefulness of the generated descriptions in validating the **DSA** ontology

7.1.2. How does the semantic refinement helpful in validating the correctness of ontology axioms?

Fig 4 and 5 show the statistics to determine the usefulness of the generated descriptions in validating the correctness of two domain ontologies where, as before, the dotted-bars represent the ratings of the descriptions that are generated from the proposed approach and the stripped-bars denote rating of the descriptions generated by the traditional approach. Usefulness of the generated descriptions in validating the correctness of an ontology are obtained by looking at the number of descriptions which are marked as 'Cannot be determined'. The three options: Valid, Invalid and Don't know, imply that the text is useful in getting into a conclusion, whereas the option 'Cannot be determined'indicates that there is some problem in the representation. From Fig 4 and Fig 5, in case of the proposed approach, only 7 out of 41 descriptions from PD ontology and 4 out of 31 descriptions from DSA ontology were not useful in determining the quality of the ontology, whereas in case of the traditional approach, approximately 50 percentage of the descriptions were not helpful. This clearly indicates that, verbalization after semantic-refinement is more effective in applications such as ontology validation.

Statistical analysis. We observed that only $7 + 4$ out of $41 + 31$ verbalized texts are not found to be useful (i.e., *not useful*) in determining the quality of the ontology, whereas a large percentage (80%) are *useful*. This is the hypothesis which needs to be tested.

Here also we can use a chi-square test to find out how closely observed data fits our expected data. The data collected for the test is summarized in Table 12. We consider the null hypothesis as "80% of the verbalized texts are useful in knowledge validation". Correspondingly the alternate hypothesis is that "not 80% of the verbalized texts are useful in knowledge validation".

The chi-square score is obtained as 0.966 (P-value = 0.32574922), and the null hypothesis is not rejected (P-value > 0.05).

Table 12
Summary of observed and expected data for statistical analysis – finding usefulness

Category	Observed	Expected
Useful	61	57.6
Not useful	11	14.4

7.1.3. Discussion and future work

In this paper, we have tried to define the notion of redundancies in a label-set in terms of concept (both atomic as well as complex) and role hierarchies. However, the notion of redundancy is, to some extent, subjective. That is, depending on the readers' domain knowledge, the level of redundancy in the text varies. In the process of semantic-refinement, we have removed the generic information from the label-set with an assumption that the human readers would be familiar with the explicit relationships between the domain entities. In that sense, a reader with poor domain knowledge would miss out generic concept information due to the refinement process. This would be easily visible when the concept hierarchies are reduced to the specific ones alone. One possible way to overcome this problem would be by including (not all but) potentially relevant concept names, that were previously omitted in the semantic-refinement process, in the refined label-set. For example, in Table 10, we can further generalize the description of the concept "mite damage", by including additional generic concept details, as "*Mite Damage:* is a pest damage and a biotic-disorder, having at least one mite pest and all its factors are mite pests." Since only a generic concept name is included in addition to all the refined concepts, the meaning of the description is not affected. More investigation and empirical study related to this would have to be done as a future endeavor.

Another interesting method (which is not addressed in this paper) to improve the description of individuals is by considering the property assertions along with the label-sets to generate descriptions. Considering property relationships/assertions is important because validation of an ontology also involves verifying the truthfulness of the property assertions in it. In future, we would be addressing this issue by making use of the edge-label-sets (label-sets for pairs of individuals – see Section 4.1), and mapping them to the respective constraint(s) in the node-label-set of the first individual. For e.g., $\mathcal{L}_{\mathcal{O}}(a) = \{C_1, C_2, \exists hasFriend.C_3\}$, and $\mathcal{L}_{\mathcal{O}}(a, b) = \{hasFriend\}$, then $hasFriend$ in $\mathcal{L}_{\mathcal{O}}(a, b)$ can be mapped to $\exists hasFriend.C_3$ in $\mathcal{L}_{\mathcal{O}}(a)$. The description of a can be generated as "a: is a C_1 and C_2, and having b as a friend and having some C_3 as friend." An example from Table 10 would be: the description of "Black Chaff" can be improved as " *Black Chaff:* is a plant disease, having Xanthomonas sp as a factor, having at least one bacteria and all its factors are bacterium." It would be our future work to further look at how to remove the redundancies in the improved descriptions – in the mentioned example, "having Xanthomonas sp as a factor" and "having at least one bacteria", can be further combined, since "Xanthomonas sp" is an individual of the concept Bacteria.

Another future work is related to ontology validation application. According to the domain experts, a persisting problem with any validation phase (especially when it involves descriptions of ontology entities and experts validating the verbalized knowledge) is that, when the ontology becomes very large and complex, validation phase becomes a bottleneck for the entire development cycle. One way to overcome this issue is by considering only a relevant subset of individuals and concepts and their descriptions for validity check, so that a rough estimate of the erroneous formalisms in the ontology can be identified quickly. A study on the order in which the descriptions are to be presented to an expert so that an early detection of invalid knowledge can be made possible, would be another future work. Furthermore, the presented approach could be further looked at as a method to find errors in inconsistent ontologies, which is an important problem that verbalization tools could help us with.

8. Conclusion

A novel method for verbalizing the definitions (called natural language descriptions) of ontology entities is presented in the paper. The descriptions are not merely verbatim translations of logical axioms of the ontology. Instead, they are generated from the set of logical restrictions satisfied by individuals and concepts of the ontology on which semantic simplification had been carried out. We propose an inference-based refinement approach for this purpose. We find that the proposed method indeed gives redundancy-free descriptions of individuals and concepts.

Our time-budgeted empirical studies based on two ontologies have shown that the redundancy-free description of the domain knowledge is helpful in understanding the formalized knowledge more effectively and also useful in validating them, typically for humans who are familiar with the domain semantics.

Acknowledgements

This project is funded by Ministry of Human Resource Development, Gov. of India. We express our fullest gratitude to the participants of our evaluation process: Dr. S.Gnanasambadan (Director of Plant Protection, Quarantine & Storage), Ministry of Agriculture, Gov. of India; Mr. J. Delince and Mr. J. M. Samraj, Department of Social Sciences AC & RI, Killikulam, Tamil Nadu, India; Ms. Deepthi.S (Deputy Manager), Vegetable and Fruit Promotion Council Keralam (VFPCK), Kerala, India; Dr. K.Sreekumar (Professor) and students, College of Agriculture, Vellayani, Trivandrum, Kerala, India. We also thank all the un-

dergraduate and post-graduate students of Indian Institute of Technology, Madras, who have participated in the empirical study.

References

[1] Ion Androutsopoulos, Gerasimos Lampouras, and Dimitrios Galanis. Generating natural language descriptions from OWL ontologies: the naturalowl system. *CoRR*, abs/1405.6164, 2014.

[2] Franz Baader, Diego Calvanese, Deborah L. McGuinness, Daniele Nardi, and Peter F. Patel-Schneider, editors. *The description logic handbook: theory, implementation, and applications*. Cambridge University Press, New York, NY, USA, 2003.

[3] Anne Cregan, Rolf Schwitter, and Thomas Meyer. Sydney owl syntax - towards a controlled natural language syntax for owl 1.1. In Christine Golbreich, Aditya Kalyanpur, and Bijan Parsia, editors, *OWLED*, volume 258 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2007.

[4] Vania Dimitrova, Ronald Denaux, Glen Hart, Catherine Dolbear, Ian Holt, and Anthony G. Cohn. Involving domain experts in authoring OWL ontologies. In *The Semantic Web - ISWC 2008, 7th International Semantic Web Conference, ISWC 2008, Karlsruhe, Germany, October 26-30, 2008. Proceedings*, pages 1–16, 2008.

[5] Astrid Duque-Ramos, Manuel Quesada-Martínez, Miguela Iniesta-Moreno, Jesualdo Tomás Fernández-Breis, and Robert Stevens. Supporting the analysis of ontology evolution processes through the combination of static and dynamic scaling functions in oquare. *Journal of Biomedical Semantics*, 7(1):63, Oct 2016.

[6] Stephan Grimm and Jens Wissmann. Elimination of redundancy in ontologies. In *Proceedings of the 8th Extended Semantic Web Conference on The Semantic Web: Research and Applications - Volume Part I*, ESWC'11, pages 260–274, Berlin, Heidelberg, 2011. Springer-Verlag.

[7] Glen Hart, Catherine Dolbear, and John Goodwin. Lege feliciter: Using structured english to represent a topographic hydrology ontology. In Christine Golbreich, Aditya Kalyanpur, and Bijan Parsia, editors, *OWLED*, volume 258 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2007.

[8] Daniel Hewlett, Aditya Kalyanpur, Vladimir Kolovski, and Christian Halaschek-wiener. Effective nl paraphrasing of ontologies on the semantic web. In *End User Semantic Web Interaction Workshop (ISWC 2015)*, 2005.

[9] Ian Horrocks, Ulrike Sattler, and Stephan Tobies. Reasoning with individuals for the description logic shiq. *CoRR*, cs.LO/0005017, 2000.

[10] Mustafa Jarrar, C. Maria, and Keet Paolo Dongilli. Multilingual verbalization of orm conceptual models and axiomatized ontologies. Technical report, 2006.

[11] Kaarel Kaljurand. *Attempto Controlled English as a Semantic Web Language*. PhD thesis, Faculty of Mathematics and Computer Science, University of Tartu, 2007.

[12] Kaarel Kaljurand and Norbert E Fuchs. Verbalizing owl in attempto controlled english. In *OWLED*, volume 258, 2007.

[13] Silvio Peroni. A simplified agile methodology for ontology development. In *OWLED*, 2016.

[14] I.O. Sanya and E.M. Shehab. A framework for developing engineering design ontologies within the aerospace industry. *International Journal of Production Research*, 53(8):2383–2409, 2015.

[15] M. Schmidt-Schau and G. Smolka. Attributive concept descriptions with complements. *Artificial Intelligence*, 48(1):1–26, 1991.

[16] Robert Stevens, James Malone, Sandra Williams, Richard Power, and Allan Third. Automating generation of textual class definitions from owl to english. *J. Biomedical Semantics*, 2(S-2):S5, 2011.

[17] Allan Third. "hidden semantics": What can we learn from the names in an ontology? In *Proceedings of the Seventh International Natural Language Generation Conference*, INLG '12, pages 67–75, Stroudsburg, PA, USA, 2012. Association for Computational Linguistics.

[18] Allan Third, Sandra Williams, and Richard Power. Owl to english : a tool for generating organised easily-navigated hypertexts from ontologies. 2011.

Appendix A

Proofs for the rules in the rule-sets 5 to 7

Here we use proof-by-contradiction as the proof method. Given a rule of the form $P \equiv Q$, we prove $P \sqsubseteq Q \sqcap Q \sqsubseteq P$, by negating it and proving $(P \sqcap \neg Q) \sqcup (Q \sqcap \neg P)$ as false.

Consider that all the following rules are defined on an ontology $\mathcal{O}$ with R and S as its roles, and U and V are two of its concepts.

Rule 5a: Given the ontology $\mathcal{O}$, $\exists R.U \sqcap \forall R.U \equiv \Im R.U$. The proof is trivial, and can be easily derived from the definition of $\Im R.U$.

Rule 5b: If $\mathcal{O} \models U \sqsubseteq V, S \sqsubseteq R$, then for $\forall R.U \sqcap \exists S.V \equiv \Im R.U \sqcap \Im S.U$.

Assume that $\forall R.U \sqcap \exists S.V \sqcap \neg(\Im R.U \sqcap \Im S.U)$ is true. We can write it as: $\forall R.U \sqcap \exists S.V \sqcap \neg((\exists R.U \sqcap \forall R.U) \sqcap (\exists S.U \sqcap \forall S.U)) \equiv \forall R.U \sqcap \exists S.V \sqcap (\forall R.\neg U \sqcup \exists R.\neg U \sqcup \forall S.\neg U \sqcup \exists S.\neg U) \implies \forall R.U \sqcap \forall S.U \sqcap \exists S.V \sqcap (\forall R.\neg U \sqcup \exists R.\neg U \sqcup \forall S.\neg U \sqcup \exists S.\neg U)$ (since $S \sqsubseteq R, \forall R.U \implies \forall S.U) \equiv \forall R.U \sqcap \forall S.U \sqcap \exists S.V \sqcap (\forall R.\neg U \sqcup \exists R.\neg U \sqcup \forall S.\neg U \sqcup \exists S.\neg U) \equiv (\underline{\forall R.U} \sqcap \forall S.U \sqcap \exists S.V \sqcap \underline{\forall R.\neg U}) \sqcup (\underline{\forall R.U} \sqcap \forall S.U \sqcap \exists S.V \sqcap \underline{\exists R.\neg U}) \sqcup (\forall R.U \sqcap \underline{\forall S.U} \sqcap \exists S.V \sqcap \underline{\forall S.\neg U}) \sqcup (\forall R.U \sqcap \underline{\forall S.U} \sqcap \exists S.V \sqcap \underline{\exists S.\neg U})$, contradiction.

Now, assume that $(\exists R.U \sqcap \forall R.U \sqcap \exists S.U \sqcap \forall S.U) \sqcap \neg(\forall R.U \sqcap \exists S.V)$ is true. $\equiv \exists R.U \sqcap \forall R.U \sqcap \exists S.U \sqcap \forall S.U \sqcap (\exists R.\neg U \sqcup \forall S.\neg V) \equiv (\exists R.U \sqcap \forall R.U \sqcap$

$\exists S.U \sqcap \forall S.U \sqcap \exists R.\neg U) \sqcup (\exists R.U \sqcap \forall R.U \sqcap \exists S.U \sqcap \forall S.U \sqcap \forall S.\neg V) \equiv (\exists R.U \sqcap \underline{\forall R.U} \sqcap \exists S.U \sqcap \forall S.U \sqcap \underline{\exists R.\neg U}) \sqcup (\exists R.U \sqcap \forall R.U \sqcap \underline{\exists S.U} \sqcap \forall S.U \sqcap \forall S.\neg V \sqcap \underline{\forall S.\neg U})$, (Since, $U \sqsubseteq V$,) contradiction.

Rule 5c: If $\mathcal{O} \models V \sqsubseteq U, S \sqsubseteq R$, then for $\forall R.U \sqcap \exists S.V \equiv \Im R.U \sqcap \exists S.V$.

Assume that $\forall R.U \sqcap \exists S.V \sqcap \neg(\Im R.U \sqcap \exists S.V)$ is true. We can write it as: $\equiv \forall R.U \sqcap \exists S.V \sqcap \neg(\exists R.U \sqcap \forall R.U \sqcap \exists S.V)$ (by the deftn. of $\Im R.U$) $\equiv \forall R.U \sqcap \exists S.V \sqcap (\forall R.\neg U \sqcup \exists R.\neg U \sqcup \forall S.\neg V) \equiv \underline{\forall R.U} \sqcap \exists S.V \sqcap \underline{\forall R.\neg U} \sqcup \underline{\forall R.U} \sqcap \exists S.V \sqcap \underline{\exists R.\neg U} \sqcup \forall R.U \sqcap \underline{\exists S.V} \sqcap \underline{\forall S.\neg V.}$

Now, assume that $(\Im R.U \sqcap \exists S.V) \sqcap \neg(\forall R.U \sqcap \exists S.V)$ is true. $\equiv \forall R.U \sqcap \exists R.U \sqcap \exists S.V \sqcap (\exists R.\neg U \sqcup \forall S.\neg V) \equiv \underline{\forall R.U} \sqcap \exists R.U \sqcap \exists S.V \sqcap \underline{\exists R.\neg U} \sqcup \forall R.U \sqcap \exists R.U \sqcap \underline{\exists S.V} \sqcap \underline{\forall S.\neg V.}$

Rule 6a: If $\mathcal{O} \models U \sqsubseteq V, R \sqsubseteq S$, then for $n \geq m$, $\geq nR.U \sqcap \geq mS.V \equiv \geq nR.U$.

Assume that, $\geq nR.U \sqcap \geq mS.V \sqcap \neg(\geq nR.U)$ is true. We can write it as: $\underline{\geq nR.U} \sqcap \geq mS.V \sqcap \underline{\leq (n-1)R.U}$, Contradiction.

Now assume that, $\geq nR.U \sqcap \neg(\geq nR.U \sqcap \geq mS.V)$ is true. We can write it as: $\geq nR.U \sqcap (\leq (n-1)R.U \sqcup \leq (m-1)S.V) \equiv (\underline{\geq nR.U} \sqcap \underline{\leq (n-1)R.U}) \sqcup (\underline{\geq nR.U} \sqcap \underline{\leq (m-1)S.V})$, contradiction. In the second conjunctive clause $\geq nR.U \implies \geq nS.V$ (since $U \sqsubseteq V \& R \sqsubseteq S$) , for $n \geq m$, $\geq nR.U \sqcap \leq (m-1)S.V$ is a contradiction.

Rule 6b: If $\mathcal{O} \models V \sqsubseteq U, S \sqsubseteq R$, then for $n \geq 1$, $\exists R.U \sqcap \geq nS.V \equiv \geq nS.V$.

Assume that, $n \geq 1$, $\exists R.U \sqcap \geq nS.V \sqcap \neg(\geq nS.V)$ is true. We can write it as: $\exists R.U \sqcap \underline{\geq nS.V} \sqcap \underline{\leq (n-1)S.V}$, contradiction.

Now, assume that $\geq nS.V \sqcap \neg(\exists R.U \sqcap \geq nS.V)$ is true. We can write it as: $\geq nS.V \sqcap (\forall R.\neg U \sqcup \leq (n-1)S.V) \equiv (\underline{\geq nS.V} \sqcap \underline{\forall R.\neg U}) \sqcup (\underline{\geq nS.V} \sqcap \underline{\leq (n-1)S.V})$, contradiction. The contradiction in the first conjunctive expression is because: $\geq nS.V \implies \exists S.V \implies \exists R.U$ which contradicts with $\forall R.\neg U$.

Rule 6c: If $\mathcal{O} \models U \sqsubseteq V$, then for $n = 1$, $\exists R.U \sqcap \leq nR.V \equiv \exists_{=1} R.U \sqcap \exists_{=1} R.V$.

Assume that, $\exists R.U \sqcap \leq nR.V \sqcap \neg(\exists_{=1} R.U \sqcap \exists_{=1} R.V)$ is true. We can write it as: $\exists R.U \sqcap \leq nR.V \sqcap \neg(\exists R.U \sqcap \leq 1R.U \sqcap \exists_{=1} R.V) \equiv (\underline{\exists R.U} \sqcap \leq 1R.V \sqcap \underline{\forall R.\neg U}) \sqcup (\exists R.U \sqcap \underline{\leq 1R.V} \sqcap \underline{\geq 2R.U}) \sqcup (\exists R.U \sqcap \leq 1R.V \sqcap \neg \exists_{=1} R.V)$, contradiction. The second conjunctive clause is a contradiction because: $\leq 1R.V \implies \leq 1R.U$ (since $U \sqsubseteq V$), which contradicts with $\geq 2R.U$. In the third conjunctive expression, $\exists R.U \implies \exists R.V$, now, $\neg(\exists_{=1} R.V) \sqcap \exists R.V \implies \geq 2R.V$, which contradicts with $\leq 1R.V$.

Now, assume that $\exists_{=1} R.U \sqcap \exists_{=1} R.V \sqcap \neg(\exists R.U \sqcap \leq 1R.V)$ is true. We can write it as: $\exists_{=1} R.U \sqcap \exists_{=1} R.V \sqcap (\forall R.\neg U \sqcup \geq 2R.V) \equiv (\underline{\exists_{=1} R.U} \sqcap \exists_{=1} R.V \sqcap \underline{\forall R.\neg U}) \sqcup (\exists_{=1} R.U \sqcap \underline{\exists_{=1} R.V} \sqcap \underline{\geq 2R.V})$, contradiction.

Rule 6d: If $\mathcal{O} \models U \sqsubseteq V, R \sqsubseteq S$, then for a whole number n, $\geq nR.U \sqcap \leq nS.V \equiv \exists_{=n} R.U \sqcap \exists_{=n} S.V$.

Assume that, $\geq nR.U \sqcap \leq nS.V \sqcap \neg(\exists_{=n} R.U \sqcap \exists_{=n} S.V)$ is true. We can write it as: $\geq nR.U \sqcap \leq nS.V \sqcap \neg(\leq nR.U \sqcap \geq nR.U \sqcap \leq nS.V \sqcap \geq nS.V) \equiv \geq nR.U \sqcap \leq nS.V \sqcap (\geq (n+1)R.U \sqcup \leq (n-1)R.U \sqcup \geq (n+1)S.V \sqcup \leq (n-1)S.V) \equiv (\underline{\geq nR.U} \sqcap \leq nS.V \sqcap \underline{\geq (n+1)R.U}) \sqcup (\underline{\geq nR.U} \sqcap \leq nS.V \sqcap \underline{\leq (n-1)R.U}) \sqcup (\underline{\geq nR.U \sqcap \leq nS.V} \sqcap \underline{\geq (n+1)S.V}) \sqcup (\underline{\geq nR.U} \sqcap \leq nS.V \sqcap \underline{\leq (n-1)S.V})$, contradiction. In the third conjunctive expression, $\geq nR.U \sqcap \leq nS.V \implies \exists_{=n} S.V$, which contradicts with $\geq (n+1)S.U$.

Now, assume that $\exists_{=n} R.U \sqcap \exists_{=n} S.V \sqcap \neg(\geq nR.U \sqcap \leq nS.V)$ is true. We can write it as: $\leq R.U \sqcap \geq nR.U \sqcap \leq nS.V \sqcap nS.V \sqcap (\leq (n-1)R.U \sqcup \geq (n+1)S.V) \equiv (\leq R.U \sqcap \underline{\geq nR.U} \sqcap \leq nS.V \sqcap nS.V \sqcap \underline{\leq (n-1)R.U}) \sqcup (\leq R.U \sqcap \geq nR.U \sqcap \underline{\leq nS.V} \sqcap nS.V \sqcap \underline{\geq (n+1)S.V})$, contradiction.

Rule 7a: If $\mathcal{O} \models U \sqsubseteq V, R \sqsubseteq S$, then $\exists R.U \sqcap \exists_{=1} S.V \equiv \exists_{=1} R.U \sqcap \exists_{=1} S.V$.

Assume that, $\exists R.U \sqcap \exists_{=1} S.V \sqcap \neg(\exists_{=1} R.U \sqcap \exists_{=1} S.V)$ is true. That is, $\exists R.U \sqcap \leq 1S.V \sqcap \geq 1S.V \sqcap (\geq 2R.U \sqcup \leq 0R.U \sqcup \geq 2S.V \sqcup \geq 0S.V) \equiv (\exists R.U \sqcap \underline{\exists_{=1} S.V} \sqcap \underline{\geq 2R.U}) \sqcup (\underline{\exists R.U} \sqcap \exists_{=1} S.V \sqcap \underline{\leq 0R.U}) \sqcup (\exists R.U \sqcap \underline{\exists_{=1} S.V} \sqcap \underline{\geq 2S.V}) \sqcup (\exists R.U \sqcap \underline{\exists_{=1} S.V} \sqcap \underline{\leq 0S.V})$, Contradiction. The contradiction in the first clause is because: since $U \sqsubseteq V \& R \sqsubseteq S; \geq 2R.U \implies \geq 2S.V; \geq 2S.V$ contradicts with $\exists_{=1} S.V$.

Now assume that $\exists_{=1} R.U \sqcap \exists_{=1} S.V \sqcap \neg(\exists R.U \sqcap \exists_{=1} S.V)$ is true. We can write it as: $\exists_{=1} R.U \sqcap \exists_{=1} S.V \sqcap (\forall R.\neg U \sqcup \neg(\exists_{=1} S.V)) \equiv (\underline{\exists_{=1} R.U} \sqcap \exists_{=1} S.V \sqcap \underline{\forall R.\neg U}) \sqcup (\exists_{=1} R.U \sqcap \underline{\exists_{=1} S.V} \sqcap \underline{\neg(\exists_{=1} S.V)})$, Contradiction.

Rule 7b: If $\mathcal{O} \models U \sqsubseteq V, R \sqsubseteq S$, then $\Im R.U \sqcap \exists_{=1} S.V \equiv \exists_{=1} R.U \sqcap \exists_{=1} S.V \sqcap \Im R.U$.

Assume that, $\Im R.U \sqcap \exists_{=1} S.V \sqcap \neg(\exists_{=1} R.U \sqcap \exists_{=1} S.V \sqcap \Im R.U)$ is true.

$\equiv \exists R.U \sqcap \forall R.U \sqcap \exists_{=1} S.V \sqcap (\neg(\exists_{=1} R.U) \sqcup \neg(\exists_{=1} S.V) \sqcup \neg(\Im R.U)) \equiv (\exists R.U \sqcap \forall R.U \sqcap \underline{\exists_{=1} S.V} \sqcap \underline{\neg(\exists_{=1} R.U)}) \sqcup (\exists R.U \sqcap \forall R.U \sqcap \underline{\exists_{=1} S.V} \sqcap \underline{\neg(\exists_{=1} S.V)}) \sqcup (\underline{\exists R.U \sqcap \forall R.U} \sqcap \exists_{=1} S.V \sqcap \underline{\neg(\Im R.U)}))$, Contradiction. The contradiction in the first conjunctive clause is because: given $x \in \exists R.U \sqcap \neg(\exists_{=1} R.U)$, it implies $x \in \geq 1R.U \implies x \in \geq 1S.V$ (since $R \sqsubseteq S$ and $U \sqsubseteq V$) which contradicts with $\exists_{=1} S.V$. In the third conjunctive clause, $\neg(\Im R.U) \equiv \neg(\forall R.U \sqcap \exists R.U) \equiv \exists R.\neg U \sqcup \forall R.\neg U$, both these cases contradict with $\exists R.U \sqcap \forall R.U$.

Now assume that, $\exists_{=1} R.U \sqcap \exists_{=1} S.V \sqcap \Im R.U \sqcap \neg(\Im R.U \sqcap \exists_{=1} S.V)$ is true.

$\equiv \exists_{=1} R.U \sqcap \exists_{=1} S.V \sqcap \Im R.U \sqcap \neg(\forall R.U \sqcap \exists R.U \sqcap \exists_{=1} S.V) \equiv (\underline{\exists_{=1} R.U} \sqcap \exists_{=1} S.V \sqcap \Im R.U \sqcap \underline{\exists R.\neg U}) \sqcup (\underline{\exists_{=1} R.U} \sqcap \exists_{=1} S.V \sqcap \Im R.U \sqcap \underline{\forall R.\neg U}) \sqcup (\exists_{=1} R.U \sqcap \underline{\exists_{=1} S.V} \sqcap \Im R.U \sqcap \underline{\neg(\exists_{=1} S.V)})$), Contradiction.

Rule 7c: If $\mathcal{O} \models U \sqsubseteq V$, then $\exists_{=n} R.U \sqcap \geq mR.V \equiv \exists_{=n} R.U \sqcap \geq (m-n)R.(V \sqcup \neg U)$ for $m \geq n$.

Assuming that $\exists_{=n} R.U \sqcap \geq mR.V \sqcap \neg(\exists_{=n} R.U \sqcap \geq (m-n)R.(V \sqcup \neg U))$ is true.

$\equiv \exists_{=n} R.U \sqcap \geq mR.V \sqcap \neg(\leq nR.U \sqcap \geq nR.U \sqcap \geq (m-n)R.(V \sqcup \neg U))$

$\equiv \exists_{=n} R.U \sqcap \geq mR.V \sqcap (\geq (n+1)R.U \sqcup \leq (n-1)R.U \sqcup \leq (m-n-1)R.(V \sqcup \neg U))$

$\equiv \underline{\exists_{=n} R.U} \sqcap \geq mR.V \sqcap \underline{\geq (n+1)R.U} \sqcup \underline{\exists_{=n} R.U} \sqcap \geq mR.V \sqcap \underline{\leq (n-1)R.U} \sqcup \underline{\exists_{=n} R.U \sqcap \geq mR.V} \sqcap \leq (m-n-1)R.(V \sqcup \neg U)$

The contradictions in the first two conjunctive clauses are trivial, in the third clause, $\exists_{=n} R.U \sqcap \geq mR.V$ implies $\geq (m-n)R.(V \sqcup \neg U)$ which contradicts with $\leq (m-n-1)R.(V \sqcup \neg U)$.

Now, assume that, $\exists_{=n} R.U \sqcap \geq (m-n)R.(V \sqcup \neg U) \sqcap \neg(\exists_{=n} R.U \sqcap \geq mR.V)$ is true.

$\equiv \exists_{=n} R.U \sqcap \geq (m-n)R.(V \sqcup \neg U) \sqcap (\neg(\exists_{=n} R.U) \sqcup \leq m-1R.V)$

$\equiv (\underline{\exists_{=n} R.U} \sqcap \geq (m-n)R.(V \sqcup \neg U) \sqcap \underline{\neg(\exists_{=n} R.U)}) \sqcup (\underline{\exists_{=n} R.U \sqcap \geq (m-n)R.(V \sqcup \neg U)} \sqcap \underline{\leq (m-1)R.V})$

In the second conjunctive clause, contradiction can be found as follows: an $x \in \geq (m-n)R.(V \sqcup \neg U)$ implies x has more than $m-n$ R relations to $\neg U \sqcap V$, since $x \in \exists_{=n} R.U$, we can say that x has more than $m-n+n$ R relations to V, which can be written as $x \in \geq mR.V$. Clearly, this contradicts with $\leq (m-1)R.V$.