Ontology engineering with rough concepts and instances

C. Maria Keet

KRDB Research Centre, Free University of Bozen-Bolzano, Italy, keet@inf.unibz.it

Abstract. A scenario in ontology development and its use is hypothesis testing, such as finding new subconcepts based on the data linked to the ontology. During such experimentation, knowledge tends to be vague and the associated data is often incomplete, which OWL ontologies normally do not consider explicitly. To fill this gap, we use OWL 2 and their application infrastructures together with rough sets. Although OWL 2 QL is insufficient to represent most of rough set's semantics, the mapping layer of its Ontology-Based Data Access framework that links concepts in the ontology to queries over the data source suffice to ascertain if a concept is rough, which subsequently can be modelled more precisely in an OWL 2 DL ontology. We summarise the trade-offs and validate it with the HGT ontology and its 17GB genomics database and with sepsis, which demonstrates it is an encouraging step toward comprehensive and usable rough ontologies.

1 Introduction

It has been noted that scientist want to use ontologies together with data, such as hypothesizing that some subclass or relation exists and subsequently to validate this either in the laboratory or against the instances already represented in the knowledge base [1]. For such a putative new concept, one would want to be able to find those instances with the right combination of object and data properties, i.e., taking a 'guessed' collection of attributes that is subsequently experimentally validated against the data (e.g., [2]). Such guessing includes dealing with incomplete or otherwise *vague* data and information. Ideally, for all relevant individuals belonging to the putative concept, each value of the chosen properties is distinct, but this may not be the case due to the limited data or insufficiency of the selected properties so that some individuals are indistinguishable from each other and therewith instantiating a *rough concept* that represents the intensional aspects of the rough set. Despite the vagueness, it still can be useful in the ontology engineering process to include such a rough concept in the ontology

Hence, a *rough ontology* can be useful. To support such usage of ontologies, one needs a language with which one can represent, at least, rough concepts as the intensional representation of the corresponding rough set and a way to persistently relate the instance data to the rough concepts. Various extensions of Description Logics (DL) and OWL languages have been proposed for rough ontologies [3–8], which diverge in commitment as to which aspects of rough sets are included in the ontology language and they concern theory instead of demonstrating successful use of the rough ontology in ontology engineering. As it turns out, there is no perfect DL language, reasoner, and ontology development tool that does it all with respect to the semantics of rough sets, nor will there be if one adheres to the hard requirement of staying within the decidable

fragment of first order logic, let alone within the tractable zone. However, some results can be obtained already: in addition to representing most of rough sets' semantics with the recently standardised OWL 2 DL, the linking to data and, moreover, ascertaining if a concept is really a rough concept can be achieved within the framework of Ontology-Based Data Access (OBDA) [9] by exploiting the mapping layer. To demonstrate it is not merely theoretically possible to have rough concepts and vague instances in one's ontologies, but that it is indeed practically possible, we take the use cases about horizontal gene transfer (HGT) with a hypothesized (rough) concept Promiscuous Bacterium, and demonstrate how this can be modelled more precisely in an OWL 2 DL ontology (but not used with the data) and deployed in an OBDA system using a simpler ontology (roughly in OWL 2 QL) so that the instances from the 17GB large HGT-DB database can be retrieved using a structured process of automation and manual intervention.

The remainder of the paper is structured as follows. We summarise the theoretical assessment on the feasibility of rough ontologies in Section 2. Experimental results with rough concepts and with vague instances will be presented in Section 3 and we close with conclusions in section 4.

2 Rough concepts and rough ontology languages

To be able to have a correspondence of a rough set with a rough concept in a rough ontology and to represent its essential characteristics, we first outline the basics of rough sets following the standard "Pawlak rough set model" [10] and then summarize requirements and trade-offs to include such roughness features in DL-based OWL ontologies.

2.1 Rough sets

The Pawlak rough set model is depicted informally in Fig. 1 and formally, it is as follows. $I = (U, A)$ is called an *information system*, where U is a non-empty finite set of objects and A a finite non-empty set of attributes and such that for every $a \in A$, we have the function $a : U \mapsto V_a$ where v_a is the set of values that attribute a can have. For any subset of attributes $P \subseteq A$, one can define the equivalence relation $\text{IND}(P)$ as $\text{IND}(P) = \{(x, y) \in U \times U \mid \forall a \in P, a(x) = a(y)\}$ so that $\text{IND}(P)$ generates a partition of U, which is denoted with $U/\text{IND}(P)$. If $(x, y) \in \text{IND}(P)$, then x and y are *p-indistinguishable* with respect to the attributes in P. From the objects in universe U, we want to represent set X such that $X \subseteq U$ using the attribute set P where $P \subseteq A$. X may not be represented in a crisp way—the set may include and/or exclude objects which are indistinguishable on the basis of the attributes in P—but it can be approximated by using lower approximation, $\underline{P}X = \{x \mid [x]_P \subseteq X\}$, and upper approximation, $\overline{P}X = \{x \mid [x]_P \cap X \neq \emptyset\}$, where $[x]_P$ denotes the equivalence classes of the p-indistinguishability relation. The *lower approximation* is the set of objects that are *positively* classified as being members of set X, i.e., it is the union of all equivalence classes in $[x]_P$. The *upper approximation* is the set of objects that are *possibly* in X; its complement, $U - \overline{P}X$, is the *negative region* with sets of objects that are definitely not in X (i.e., $\neg X$). Then, "with every rough set we associate two *crisp* sets, called *lower* and *upper approximation*" [10], which is commonly denoted as a tuple $X = \langle \underline{X}, \overline{X} \rangle$.

The difference between the lower and upper approximation, $B_P X = \overline{P}X - \underline{P}X$, is the *boundary region* of which its objects neither can be classified as to be member of X nor that they are not in X; if $B_P X = \emptyset$ then X is, in fact, a crisp set with respect to P and when $B_P X \neq \emptyset$ then X is rough w.r.t. P. The *accuracy of approximation*, $\alpha_P X$, provides a measure of how well the rough set approximates the target set with respect to P, e.g., $\alpha_P X = \frac{|\underline{P}X|}{|\overline{P}X|}$. Clearly, if $\alpha_P X = 1$, then the boundary region $B_P X$ is empty and thus X is crisp. Useful for reasoning is that $\underline{P}X \subseteq X \subseteq \overline{P}X$.

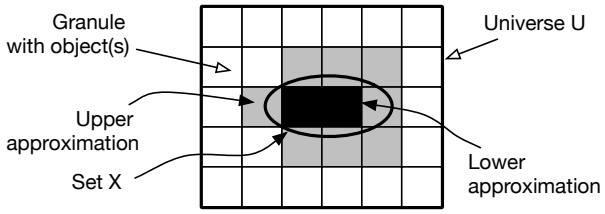

Fig. 1. A rough set and associated notions (Source: based on [10]).

2.2 Prospects for rough OWL ontologies

Due to space limitations, only the outcome of the analysis will be summarised here; a more comprehensive assessment and argumentation is described in [11].

OWL ontologies use a richer language [12] than the simple $I = (U, A)$ of rough sets, in particular concerning how to represent 'attributes' of a concept $C \in \mathcal{C}$ (with object properties $R \in \mathcal{R}$ or data properties $D \in \mathcal{D}$) and object properties' properties. Another difference is that we need a model-theoretic semantics for lower and upper approximation, $\underline{C}$ and $\overline{C}$, and rough concept, "$\wr C$", and impose that the attributes used to compute $\underline{C}$ and $\overline{C}$ are represented in the ontology.

The semantics of the approximations is straightforward, with E denoting the reflexive, symmetric and transitive indistinguishability (i.e., equivalence) relation: $\underline{C} = \{x \mid \forall y : (x, y) \in E \rightarrow y \in C\}$ and $\overline{C} = \{x \mid \exists y : (x, y) \in E \wedge y \in C\}$. Regarding rough sets' tuple notation, $X = \langle \underline{X}, \overline{X} \rangle$, there can be an analougous one for concepts, $\wr C = \langle \underline{C}, \overline{C} \rangle$, but this cannot be represented as such in the ontology. A solution to this is to 'flatten out' the tuple by relating $\underline{C}$ and $\overline{C}$ to $\wr C$ with newly introduced object properties, say, $lapr$ and $uapr$ (i.e., quantifying over *sets*, not objects that are member of the respective sets). This enables one to make explicit the knowledge about how the three concepts relate to each other: $\wr C$ is identified by the combination of its $\underline{C}$ and $\overline{C}$, which is analogous to a weak entity type in EER, i.e., for each $\wr C$, there is exactly one $\underline{C}$ and one $\overline{C}$, and vv. Last, for rough set's A, the set of 'attributes' in OWL amounts to $\mathcal{R} \cup \mathcal{D}$, and each $p_i \in P$ to compute the rough concepts must have $\wr C$ as its domain.

Overall, we now have a more precise notion of $\wr C$ cf. the tuple notation in [6], use both $\mathcal{R}$ and $\mathcal{D}$ for the 'attributes' of the concepts (cf. $\mathcal{R}$ only in [5, 8]), include the properties of the indistinguishability relation (cf. their omission in [7] or a similarity relation [3]), and adhere to proper declaration of $\underline{C}$, $\overline{C}$, and $\wr C$ in that they all have the same collection of properties from $\mathcal{R} \cup \mathcal{D}$ (cf. different sets of attributes in [8]).

Nevertheless, there are still three aspects to sort out for practical rough ontologies: the necessity to represent the indistinguishability relation E, the identity of a rough concept by its lower and upper approximation by means of identification constraints involving OWL object properties, and linking the ontology to (large amounts of) instances to ascertain if a putative rough concept is indeed rough or not. Currently, there is no single DL or OWL language that can handle all three features, or even two of the three. Narrowing the gap between 'offer and demand' by inventing a new language is impractical: real identification of $\wr C$ requires second order logic, or, as approximation, identification constraints for each rough concept, which is likely to increase the computational complexity even further. Therefore, we shall push the envelope of extant languages and tools and make concessions regarding semantics and performance in order to gain better insight into if development of a new language and corresponding tools are worth the effort and if so, which aspect should be tackled first, and to assess the amount of uptake by ontologists to experiment with rough concepts. For *practical* reasons, then, we narrow down the ontology languages to the DL-based OWL species, because they are W3C standardised languages, there are ontology development tools for them, they have several automated reasoners, and they are the DL of choice among the (bio-)ontologists. E's reflexivity, symmetry and transitivity, can be represented with OWL 2 DL [12]. Currently, $\wr C$ can only be added as a "`RoughC`" with its relations to the approximations so that it serves as human communication, i.e., there is no computational significance other than deducing $\underline{C} \sqsubseteq C \sqsubseteq \overline{C}$ based on the declared knowledge in the TBox. Then, given it is the interplay with the actual instances that is crucial for roughness, we need also a system that can handle large amounts of data, which currently locks one into *DL-Lite*/OWL 2 QL in the OBDA framework with QUONTO [9] that can represent even less of rough set's semantics (and of the subject domain) than OWL 2 DL. This issue can be counterbalanced partially by exploiting the mapping layer, although this is not ideal because it is not as transparent and maintainable as representing the subject domain semantics in the ontology, but we can assess our rough concepts and instances.

3 Experimentation with a rough ontology and vague instances

Given the aforementioned trade-offs, we will demonstrate how one can have either an ontology with rough concepts represented fairly comprehensively regarding their semantics (Experiment 2 and 3) or have it with more limited semantics but linked to the data and be able to perform the actual hypothesis testing against the data (Experiment 1). To be fair to the latest technologies for expressive OWL species, we also experiment with a more expressive ontology with little data by revisiting the sepsis experiment of [8] in Experiment 3.

3.1 Materials and Methods

Methodology. Different strategies for both experimentation and use of rough ontologies in operational ontology-driven information systems are possible. Because the link to data is crucial for rough ontologies, we commence with the OBDA approach and

subsequently move to a more expressive language. The methodological steps for the first two experiments are as follows:

1. Develop a basic ontology in OWL 2 QL or *DL-Lite*$_\mathcal{A}$ stored as an OWL file;
2. Obtain the relational database in Oracle, DB2, MySQL, or PostgreSQL;
3. Set up the OBDA system;
4. Declare the mappings between the classes and properties in the OWL ontology and SQL queries over the database in the OBDA system;
5. Find all rough concepts with respect to the data through posing ontology-mediated queries (in SPARQL or EQL-Lite), evaluating the result set, and adding the concepts to the ontology;
6. Migrate this ontology to an expressive OWL species, such as OWL 2 DL: (i) declare the semantics from the WHERE clause in the SQL query of the mapping layer as object and data properties in the ontology; (ii) add upper and lower approximations of each rough concept; (iii) add the indistinguishability object property with its properties (reflexive, symmetric, transitive); (iv) add the axioms relating the approximations to the rough concepts and vv.;
7. When the rough reasoning services are implemented, run the reasoner with the enhanced ontology to check satisfiability and consistency.

For step 6 in experiment 2, the following considerations are adhered to. Recollecting the relevant part of OWL 2's direct semantics [12]: take a vocabulary V with, among others, V_C denoting the set of classes, and its corresponding class interpretation function $\cdot^C$ that assigns to each class $C \in V_C$ a subset $(C)^C \subseteq \Delta^I$, and V_{OP} the set of object properties where $\cdot^{OP}$ is the object property interpretation function that assigns to each object property $OP \in V_{OP}$ a subset $(OP)^{OP} \subseteq \Delta^I \times \Delta^I$. We add rough concept, upper, and lower approximation such that $\wr\mathtt{C}, \overline{\mathtt{C}}, \underline{\mathtt{C}} \in V_C$, add the indistinguishability relation `Ind` over $\Delta^I \times \Delta^I$ such that $\mathtt{Ind} \in V_{OP}$ and the ontology contains the assertions: `ReflexiveObjectProperty(a:Ind)`, `SymmetricObjectProperty(a:Ind)`, and `TransitiveObjectProperty(a:Ind)`, and assign the semantics to the classes:

$$(\overline{\mathtt{C}})^I = \{x \in \Delta^I \mid \exists y \in \Delta^I, (x,y) \in \mathtt{Ind} \wedge y \in \mathtt{C}^I\} \tag{1}$$

$$(\underline{\mathtt{C}})^I = \{x \in \Delta^I \mid \forall y \in \Delta^I, (x,y) \in \mathtt{Ind} \rightarrow y \in \mathtt{C}^I\} \tag{2}$$

$$(\wr\mathtt{C})^I = (\langle \underline{\mathtt{C}}, \overline{\mathtt{C}} \rangle)^I = \langle (\underline{\mathtt{C}})^I, (\overline{\mathtt{C}})^I \rangle \tag{3}$$

which amounts to the assertions for any $\overline{\mathtt{C}}$ and $\underline{\mathtt{C}}$ in OWL 2 DL functional syntax,

`EquivalentClasses(`$\overline{\mathtt{C}}$ `ObjectSomeValuesFrom(a:Ind a:C))` and
`EquivalentClasses(`$\underline{\mathtt{C}}$ `ObjectAllValuesFrom(a:Ind a:C))`

Recollect that (3) is approximated in an ontology by adding the two properties, `uapr`, `lapr` $\in V_{OP}$ that each have $\wr\mathtt{C}$ as domain, and cardinality exactly 1: `ObjectPropertyDomain(a:upar a:`$\wr$`C)`, `ObjectPropertyDomain(a:lapr a:`$\wr$`C)`, `ObjectExactCardinality(1 a:uapr a:`$\overline{\mathtt{C}}$`)`, and `ObjectExactCardinality(1 a:lapr a:`$\underline{\mathtt{C}}$`)`. This is added to the ontology for each rough concept and its approximations.

Materials. Linking data from one or more data sources to an ontology is carried out with the OBDA system based on the theory and tools described in [9]. In short, the *semantic layer* was realised with Protégé 3.3.1 and the OBDA plugin for Protégé, with which one develops and inspects the OWL ontology and declares the mappings between

the classes and properties in the ontology and SQL queries over the data source (persistently stored in an `.obda` file). The *application layer* is the OBDA plugin for Protégé as well, which allows the user to pose the SPARQL user-queries over the ontology and inspect the query answer. The automated reasoner that takes care of unfolding and rewriting the SPARQL user-query into the SQL query over the data sources is QUONTO. The *data layer* consists of a single Oracle 10g database based on the HGT-DB database [14] that stores data about horizontal gene transfer and was kindly made available for testing purposes by its developers.

The ontology language for this particular implementation of the OBDA system is the OWL-ized DL language *DL-Lite*$_\mathcal{A}$ [9]. From an ontologist perspective, the *DL-Lite*$_\mathcal{A}$ ontology for OBDA, and in particular the HGT 'application ontology', has typical characteristics of a logic-based simple conceptual data model, which has 31 classes, 32 object properties, 61 data properties, and 108 subclass axioms that deal with organisms and their properties (such as name, amount of genes, and chromosomes) and genes with their properties (such as its function, location, and its statistics).

For the OWL2 DL representation of the HGT ontology and rough concepts, Protégé 4.0 was used with Fact++. The domain knowledge for Experiment 3 is taken from [8] and also required OWL 2 DL. Protégé 4.0 with Pellet 2.0 and FaCT++, and Racer Pro Preview 2.0 were used and the experiments were carried out on a Macbook Pro with Mac OS X v 10.5.8 with 2.93 GHz Intel core 2 Duo and 4 GB memory.

The supplementary files—ontologies, mappings, queries, and data—are available online through `http://obda.inf.unibz.it/obdahgtdb/obdahgtdb.html`.

3.2 Results

The setting for the first and second experiment about HGT is as follows. A geneticist has an idea about what a "promiscuous bacterium" is because some bacteria transfer and receive much more genes from other bacteria than others do. It is not fully understood who they are and why this is the case, hence the first step is to analyse the data using properties that indicate a certain promiscuity so as to find bacteria with comparatively many anomalous (foreign) DNA in their chromosome.

Experiment 1 (Promiscuous bacteria in OBDA) We specify a first attempt for representing the notion of PromiscuousBacterium in the ontology as a subtype of Organism, so that it must have more than 5 so-called FlexibleHGTgeneClusters (a set of overlapping, adjacent or nearby genes on the chromosome that are horizontally transferred) and the Percentage of genes on the chromosome that are predicted to be horizontally acquired as > 10. Given the limitations of the language and to not have the intended meaning of PromiscuousBacterium (as in (9), below) partially in the ontology and partially in the mapping layer, we chose to put all properties in the OBDA mapping layer; that is, we have PromiscuousBacterium $\sqsubseteq$ Organism in the OWL ontology, and then made a mapping between PromiscuousBacterium in the ontology and an SQL query over the relational database (see Fig. 2): the head of the mapping with the class in the ontology and those attributes in the database with which we identify a promiscuous bacterium, is shown with the line `PromiscuousBacterium(getPromBact($abbrev,$ccount, $percentage))` under mapping `M:0` and the mapping body, i.e., SQL query over the

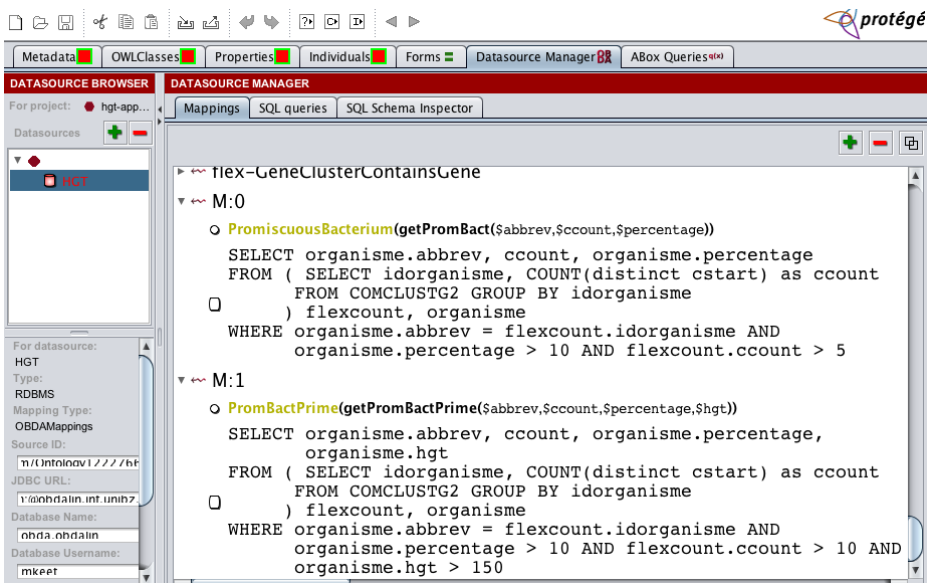

Fig. 2. Mappings for PromiscuousBacterium and its subclass PromBactPrime in the OBDA plugin.

database, below it. Querying the database through the ontology with a SPARQL query using the OBDA Plugin for Protégé and answered using QUONTO, 98 objects are retrieved where *Dehalococcoides CBDB1* (`dehaloc`) and *Thermotoga maritima* (`tmar`) are truly indistinguishable bacteria, i.e. they have the same values for all the selected and constrained attributes. A few others are very close to being so, such as *Pelodictyon luteolum DSM273* (`plut`) and *Synechocystis PCC6803* (`synecho3`) who have both 6 clusters and 10.1% and 10.2%, respectively, which, practically, still lie within the error-margin of genomics data and its statistics; see online material for the full result set. Thus, by virtue of `dehaloc` and `tmar`, PromiscuousBacterium is a rough concept.

To improve the accuracy and examine if we can turn a subconcept of PromiscuousBacterium into a crisp one, a *new* data property about the number of predicted genes that are horizontally transferred—NrPredHGTgenes with integer values, set to >150—is added and the second attribute set at >10 gene clusters, which thus *revises* the assumption of what a promiscuous bacterium really is, i.e., we have a "PromBactPrime" in the ontology such that PromBactPrime $\sqsubseteq$ PromiscuousBacterium. The head and body of the mapping are depicted under `M:1` in Fig. 2. The query answer has only 89 objects and this change eliminated the indistinguishability of `dehaloc` and `tmar`, hence PromBactPrime is a *crisp* concept with respect to the data stored in the database. $\diamondsuit$

Experiment 2 (Promiscuous bacteria in OWL 2 DL) In contrast to the OBDA setting in Experiment 1, it is possible to represent the subject domain semantics of promiscuous bacteria in the OWL 2 DL ontology itself instead of in the mapping layer. The PromiscuousBacterium is a subclass of Organism in the HGT ontology with an additional object- and a data property, so that it must have more than 5 flexible hgt-gene

clusters (FlexibleHGTgeneCluster) and the percentage of genes on the chromosome that are predicted to be horizontally acquired, Percentage, greater than 10. In compact DL notation, we then have:

$$\begin{aligned}\mathsf{PromiscuousBacterium} \equiv \mathsf{Organism} \sqcap \exists\, \mathsf{Percentage.real}_{>10} \sqcap \\ \geq 6\ \mathsf{hasHGTCluster.FlexibleHGTGeneCluster}\end{aligned} \tag{4}$$

In addition, we have to add the assertions regarding the indistinguishability relation `Ind` (see Section 3.1 for relational properties) and that Promiscuous Bacterium has exactly one lower and one upper approximation, PromBactLapr and PromBactUapr, as follows:

$$\mathsf{PromiscuousBacterium} \sqsubseteq = 1\ \mathsf{lapr.PromBactLapr} \tag{5}$$

$$\mathsf{PromiscuousBacterium} \sqsubseteq = 1\ \mathsf{uapr.PromBactUapr} \tag{6}$$

$$\mathsf{PromBactLapr} \equiv \forall\ \mathsf{Ind.PromBact} \tag{7}$$

$$\mathsf{PromBactUapr} \equiv \exists\ \mathsf{Ind.PromBact} \tag{8}$$

Given that we already know from Experiment 1 that PromiscuousBacterium is indeed a rough concept and that the proposed refinement, i.e., PromBactPrime, is not, the additional crisp concept requires only the full definition without relating it to a lower and upper approximation:

$$\begin{aligned}\mathsf{PromBactPrime} \equiv \mathsf{PromiscuousBacterium} \sqcap \exists\, \mathsf{Percentage.real}_{>10} \sqcap \\ \geq 11\ \mathsf{hasHGTCluster.FlexibleHGTGeneCluster} \sqcap \\ \exists\, \mathsf{NrPredHGTgenes.integer}_{>150}\end{aligned} \tag{9}$$

Querying or instance classification with this OWL 2 DL version and the HGT data, however, is currently not feasible. ♢

Experiment 3 (Revisiting septic patients) Patients may be septic or are certainly septic, according to the so-called *Bone criteria* and Bone criteria together with three out of another five criteria, respectively. For instance, the Bone criteria (from [8]) can be encoded in OWL as being an `EquivalentClass` to `BoneSeptic`, as follows:

```
(hasDiagnosis some Infection and
hasSymptom some (Hypoperfusion or Hypotension or OrganDysfunction)
and (((temperature some int[> 38] or temperature some int[<36])
  and (respiratoryRate some int[>20] or paco2count some int[<32]))
  or ((temperature some int[>38] or temperature some int[<36]) and
       heartRate some int[>90])
  or ((temperature some int[>38] or temperature some int[<36]) and
   (leukocyteCount some int[<4000] or leukocyteCount some int[>12000]))
  or ((respiratoryRate some int[>20] or paco2count some int[<32]) and
       heartRate some int[>90])
  or ((respiratoryRate some int[>20] or paco2count some int[<32]) and
   (leukocyteCount some int[<4000] or leukocyteCount some int[>12000]))
  or (heartRate some int[>90] and (leukocyteCount some int[<4000] or
   leukocyteCount some int[>12000]))))
```

The respective encodings in Protégé 4.0 and RacerPro 2.0 preview are available online as supplementary material, as well as data of 17 'patients' such that there are indistinguishable patients and the boundary region is not empty. Protégé 4.0 with Pellet 2.0 did not work at all. Protégé 4.0 with FaCT++ works well with a few dummy concepts and a few instances, but the esoteric definitions for septic appeared to be more challenging: it crashed with an encoding including `Ind` and, with or without `Ind`, upon saving and reopening the `.owl` file, it had reordered the braces in the definition in such a way as to change its meaning so that it does not classify all 17 individuals correctly.

These observations are probably due to the fact that the software used is still in the early stages. RacerPro 2.0 Preview never crashed during exerimentation and did return the correct classifications within about 2 hours (the slow response time is likely due to the heavy use of data properties with the concrete domains; Volker Haarslev, personal communication). While the latter is an encouraging result because it works with the real definitions and a small data set, the automated reasoning clearly does not scale to [8]'s thousands of patients. $\diamondsuit$

3.3 Discussion

While a rough ontology such as the amended HGT ontology in OWL 2 DL did provide a more comprehensive and transparent way of representing the declarative knowledge of putative and actual rough concepts, it was only with the less expressive OWL 2 QL-based OBDA-enabled system that it could be experimentally validated against the data and that it was *certain* that there was no need to add approximations for PromBactPrime in the ontology. Taken together, the ontologies, OBDA, and ontology development tools with their respective reasoners provide a means to represent the steps of successive de-vaguening during experimentation, they make the selected properties explicit, and, if desired, one can keep both ≀PromiscuousBacterium and PromBactPrime in the ontologies without generating inconsistencies. The two-step process with OBDA and OWL 2 DL might seem cumbersome, but is an advance with respect to traceability compared to separately querying the database with undocumented one-off queries and adding the obtained knowledge to the OWL 2 DL ontology.

The feasibility to extend languages or tools to reflect the representation of the roughness semantics and the experimentation in ontology engineering using a reasoner, is theoretically, and thus also practically, limited. As last resort, one can add a RoughC $\sqsubseteq \top$ and let all rough concepts in the subject domain be subsumed by RoughC so that the modeller informally can keep track of the rough concepts in the ontology. For OWL 2 DL, one can also define an extension as outlined in Section 3.1, which is similar to that of [6]'s $\mathcal{RDL}_{AC}$. However, there is little to gain in automated reasoning due to scalability issues as illustrated in the experiments. The added value of such extensions may be questioned and the following two avenues will yield more usefulness from a knowledge engineering perspective.

The first direction where improvements may be achieved, is to partition the data source and import only data into the ABox of one or a few objects at a time, carry out the instance classification, export the results, merge the results after each classification step, and then assess the results; however, one-off scripting is not ideal. Alternatively, a sophisticated modularization of both the ontology and the data(base) so as to execute the reasoning only on small sections of the ontology and database (in the direction of, e.g., [15]) might be an option in the near future. The second option concerns usability: although rough ontologies work as proof-of-concept, the procedure to carry it out is not user-friendly. One may be able to turn into a feature the interaction between the more precise representation in OWL 2 DL and the linking to data with OWL 2 QL (or a similar tractable language) by upgrading it to a named scientific workflow. This guides the developer to carry out in a structured, traceable, and repeatable manner the tasks to experiment with ontologies, the associated data, and the rough concepts.

4 Conclusions

Extending OWL ontologies with the core notions of rough sets revealed both theoretical and practical challenges. Given rough sets' semantics, there is no, nor will there be, a DL-based OWL species that can represent all these aspects precisely, although expressive languages, such as OWL 2 DL, come close and some tools, such as RacerPro, can handle complex rough concept descriptions with a small amount of data. However, it is the interaction with large amounts of data that makes any extension with roughness interesting and necessary to find rough concepts, which is possible with the Ontology-Based Data Access framework. Validation of the theoretical assessment was carried out through experimentation with rough concepts and vague instances using the HGT case study and sepsis. Current and future work pertains to looking further into modularization to achieve a platform for hypothesis-driven usage of rough ontologies that will reap the greatest benefits to meet the users' requirements.

Acknowledgements I thank Umberto Straccia, Ferdinando Bobillo, and Mariano Rodríguez-Muro for feedback during the experimentation.

References

1. Keet, C.M., et al.: A survey of requirements for automated reasoning services for bio-ontologies in OWL. In: Proc. of OWLED'07. Volume 258 of CEUR-WS. (2007)
2. Marshall, M.S., et al.: Using semantic web tools to integrate experimental measurement data on our own terms. In: Proc. of KSinBIT'06. Volume 4277 of LNCS., Springer (2006) 679–688
3. Bobillo, F., Straccia, U.: Supporting fuzzy rough sets in fuzzy description logics. In: Proc. of ECSQARU'09. Volume 5590 of LNCS., Springer (2009) 676–687
4. Fanizzi, N., et al.: Representing uncertain concepts in rough description logics via contextual indiscernibility relations. In: Proc. of URSW'08. Volume 423 of CEUR-WS. (2008)
5. Ishizu, S., et al.: Rough ontology: extension of ontologies by rough sets. In: Proc. of HCI'07. Volume 4557 of LNCS., Springer (2007) 456–462
6. Jiang, Y., Wang, J., Tang, S., Xiao, B.: Reasoning with rough description logics: An approximate concepts approach. Inform. Sciences **179** (2009) 600–612
7. Liau, C.J.: On rough terminological logics. In: Proceedings of the 4th International Workshop on Rough Sets, Fuzzy Sets and machine Discovery (RSFD'96). (1996) 47–54
8. Schlobach, S., Klein, M., Peelen, L.: Description logics with approximate definitions—precise modeling of vague concepts. In: Proc. of IJCAI'07, AAAI Press (2007) 557–562
9. Calvanese, D., et al.: Ontologies and databases: The DL-Lite approach. In: Semantic Technologies for Informations Systems. Volume 5689 of LNCS., Springer (2009) 255–356
10. Pawlak, Z., Skowron, A.: Rudiments of rough sets. Inform. Sciences **177**(1) (2007) 3–27
11. Keet, C.M.: On the feasibility of description logic knowledge bases with rough concepts and vague instances. In: Proc. of DL'10. CEUR-WS (2010) 314–324 Waterloo, Canada.
12. Motik, B., Patel-Schneider, P.F., Cuenca-Grau, B.: OWL 2 web ontology language: Direct semantics. W3c recommendation, W3C (27 October 2009 2009)
13. Motik, B., et al.: OWL 2 Web Ontology Language Profiles. W3c recommendation, W3C (27 Oct. 2009) http://www.w3.org/TR/owl2-profiles/.
14. Garcia-Vallvé, S., et al.: HGT-DB: a database of putative horizontally transferred genes in prokaryotic complete genomes. Nucleic Acids Research **31**(1) (2003) 187–189
15. Lutz, C., Toman, D., Wolter, F.: Conjunctive query answering in the description logic EL using a relational database system. In: Proc. of IJCAI'09, AAAI Press (2009)