Triplifying Equivalence Set Graphs

Luigi Asprino[1,2], Wouter Beek[3], Paolo Ciancarini[2], Frank van Harmelen[3], and Valentina Presutti[1]

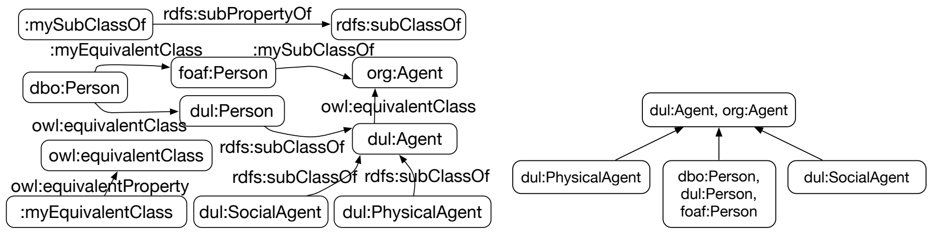
[1] STLab, ISTC-CNR, Rome, Italy
luigi.asprino@istc.cnr.it valentina.presutti@cnr.it
[2] University of Bologna, Bologna, Italy
paolo.ciancarini@unibo.it
[3] Dept. of Computer Science, VU University Amsterdam, NL
{w.g.j.beek, frank.van.harmelen}@vu.nl

Abstract. In order to conduct large-scale semantic analyses, it is necessary to calculate the deductive closure of very large hierarchical structures. Unfortunately, contemporary reasoners cannot be applied at this scale, unless they rely on expensive hardware such as a multi-node in-memory cluster. In order to handle large-scale semantic analyses on commodity hardware such as regular laptops we introduced [1] a novel data structure called Equivalence Set Graph (ESG). An ESG allows to specify compact views of large RDF graphs thus easing the accomplishment of statistical observations like the number of concepts defined in a graph, the shape of ontological hierarchies etc. ESGs are built by a procedure presented in [1] that delivers graphs as a set of maps storing nodes and edges. In this demo paper *(i)* we show how facts entailed by an ESG and the graph itself can be specified in RDF following a novel introduced ontology; and, *(ii)* we present two datasets resulting from the triplification of two ESG graphs (one for classes and one for properties).

Keywords: Semantic Web · Linked Open Data · Empirical Semantics.

1 Introducing Equivalence Set Graphs

An Equivalence Set Graph (ESG) is a tuple $\langle \mathcal{V}, \mathcal{E}, p_{eq}, p_{sub}, p_e, p_s \rangle$. The nodes $\mathcal{V}$ of an ESG are equivalence sets of terms from the universe of discourse. The directed edges $\mathcal{E}$ of an ESG are specialization relations between those equivalence sets. p_{eq} is an equivalence relation that determines which equivalence sets are formed from the terms in the universe of discourse. p_{sub} is a partial order relation that determines the specialization relation between the equivalence sets. In order to handle equivalences and specializations of p_{eq} and p_{sub} (see below for details and examples), we introduce p_e, an equivalence relation over properties (e.g., `owl:equivalentProperty`) that allows to retrieve all the properties that are equivalent to p_{eq} and p_{sub}, and p_s which is a specialization relation over



(a) RDF Knowledge Graph (b) Equivalence Set Graph

Fig. 1: An example of an RDF Knowledge Graph and its corresponding Equivalence Set Graph.

properties (e.g., `rdfs:subPropertyOf`) that allows to retrieve all the properties that specialize p_{eq} and p_{sub}.

The inclusion of the parameters p_{eq}, p_{sub}, p_e, and p_s makes the Equivalence Set Graph a very generic concept. By changing the equivalence relation (p_{eq}), ESG can be applied to classes (`owl:equivalentClass`), properties (`owl:equivalentProperty`), or instances (`owl:sameAs`). By changing the specialization relation (p_{sub}), ESG can be applied to class hierarchies (`rdfs:subClassOf`), property hierarchies (`rdfs:subPropertyOf`), or concept hierarchies (`skos:broader`).

Figure 1 shows an example of an RDF Knowledge Graph (Subfigure 1a). The equivalence predicate (p_{eq}) is `owl:equivalentClass`; the specialization predicate (p_{sub}) is `rdfs:subClassOf`, the property for asserting equivalences among predicates (p_e) is `owl:equivalentProperty`, the property for asserting specializations among predicates (p_s) is `rdfs:subPropertyOf`. The corresponding Equivalence Set Graph (Subfigure 1b) contains four equivalence sets. The top node represents the agent node, which encapsulates entities in DOLCE and W3C's Organization ontology. Three nodes inherit from the agent node. Two nodes contain classes that specialize `dul:Agent` in the DOLCE ontology (i.e. `dul:PhysicalAgent` and `dul:SocialAgent`). The third node represents the person concept, which encapsulates entities in DBpedia, DOLCE, and FOAF. The equivalence of these classes is asserted by `owl:equivalentClass` and `:myEquivalentClass`. Since `foaf:Person` specialises `org:Agent` (using `:mySubClassOf` which specialises `rdfs:subClassOf`) and `dul:Person` specialises `dul:Agent` the ESG contains an edge between the person and the agent concept.

The procedure for computing an Equivalence Set Graph has been presented in [1]. In the remaining of this document we present how to specify statements entailed by an ESG and the graph itself in RDF.

2 Equivalence Set Graph Ontology

Figure 2 depicts the ontology for specifying Equivalence Set Graphs in RDF. The prefix `esgs:` is associated with the value `https://w3id.org/edwin/ontology/`. The solution for modeling ESGs is fairly simple. An ESG is specified as an

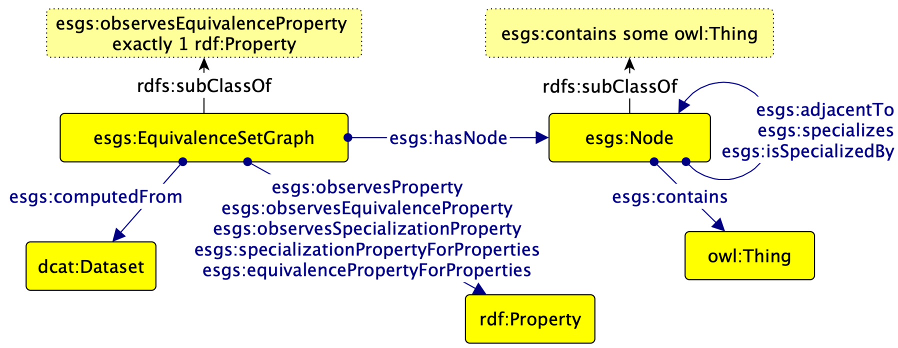


Fig. 2: The diagram of the Equivalence Set Graph Ontology expressed with the Graffoo notation.

individual of the class `esgs:EquivalenceSetGraph` and it is connected to its nodes by the object property `esgs:hasNode` having `esgs:Node` as range. Nodes are associated with entities composing the equivalence set by the object property `esgs:contains`. Edges of the graph can be specified as triples having `esgs:isAdjacentTo` (or one of its sub-properties) as predicate. Currently the ontology declares two sub-properties of `esgs:isAdjacentTo`, namely `esgs:specializes` and its inverse `esgs:isSpecializedBy` (that have been used for the analysis presented in [1]), but `esgs:isAdjacentTo` can be furtherly specialized whenever the framework is extended to allow the analysis of other kinds of relations. Individuals of `esgs:EquivalenceSetGraph` are also associated with the relations used for building the graph, namely p_{eq}, p_{sub}, p_e and p_s, by means of properties `esgs:observesEquivalenceProperty`, `esgs:observesSpecializationProperty`, `esgs:equivalencePropertyForProperties` and `esgs:specializationPropertyForProperties`. Moreover, the property `esgs:computedFrom` allows to associate an ESG with the `dcat:Dataset` from which the graph has been computed.

Rules. The ontology also defines two rules (in SWRL) that allow to materialise statements entailed by an ESG:

Equivalence Closure. If equivalence sets of a graph *?g* have been formed using the property *?peq* as ground term and two entities *?e1* and *?e2* belong to the same node *?n* (i.e. *?e1* and *?e2* belong to the equivalence set) of the graph *?g*, then, *?e1* and *?e2* are declared equivalent by means of property *?peq*.

$$\text{esgs:hasNode(?g, ?n)} \wedge \text{esgs:observesEquivalenceProperty(?g, ?peq)} \wedge \\ \text{esgs:contains(?n, ?e1)} \wedge \text{esgs:contains(?n, ?e2)} \Rightarrow \\ \text{swrlb:add(?e1, ?peq, ?e2)} \wedge \text{swrlb:add(?e2, ?peq, ?e1)}$$

Specialization. If the following conditions hold: *(i)* specialization relations (i.e. edges) of an ESG *?g* have been computed using a property *?psub* as ground term; *(ii)* nodes *?n1* and *?n2* contain the entities *?e1* and *?e2* respectively; and, *(iii)* *?n1* specializes *?n2*; then, we can assert that *?e1* specializes *?e2* by means of property *?psub*.

$$\text{esgs:observesSpecializationProperty(?g, ?psub)} \wedge \text{esgs:hasNode(?g, ?n1)} \wedge$$
$$\text{esgs:hasNode(?g, ?n2)} \wedge \text{esgs:contains(?n1, ?e1)} \wedge \text{esgs:contains(?n2, ?e2)} \wedge$$
$$\text{esgs:specializes(?n1, ?n2)} \Rightarrow \text{swrlb:add(?e1, ?psub, ?e2)}$$

3 Planned Demonstration

We plan a live demonstration where the audience will be able to compute and query a number of Equivalence Set Graphs specified in RDF. Some of Equivalence Set Graphs available for querying will be pre-computed, while others will be computed live during the demo. Pre-computed datasets include the triplication of two (very large) Equivalence Set Graphs computed from LOD-a-lot [2] and presented in [1]: one for classes and one for properties.

Moreover, the audience will be able to compute some (small) Equivalence Set Graphs using the framework presented in [1]. The framework provides a simple command line interface that allows the users to input the parameters needed for computing the ESG, i.e.: an equivalence relation (i.e. p_{eq}) and a specialization relation (i.e. p_{sub}) they would like observe (p_e and p_s will be fixed to `owl:equivalentProperty` and `rdfs:subPropertyOf`, otherwise an ESG for p_e and p_s would need to be calculated to retrieve properties in the closure of p_{eq} and p_{sub}). Then, the framework will estimate the time needed for computing the ESG for the input properties so to ensure that the ESG can be computed during the demo session. Once the ESG is computed, it will be uploaded to a triple store and available for querying.

In [1] we have shown how Equivalence Set Graphs can be used for performing statistical observation on modeling style and semantic structure of very large datasets. In this demonstration our objective is to show that Equivalence Set Graphs are also useful for retrieving data that are not explicitly asserted in an input dataset (that could be very large). By querying the triplified ESG a user is able to retrieve all the entities (e.g. classes, properties, individuals) that are implicitly equivalent to or specialized by a given entity. This is can be done even for very large datasets, like LOD-a-lot, without running reasoners or using property paths that may require expensive computational resources.

References

[1] Luigi Asprino, Wouter Beek, Paolo Ciancarini, Frank van Harmelen, and Valentina Presutti. "Observing LOD using Equivalent Set Graphs: it is mostly flat and sparsely linked". In: *Proc of ISWC 2019*.

[2] Javier D. Fernández, Wouter Beek, Miguel A. Martínez-Prieto, and Mario Arias. "LOD-a-lot - A Queryable Dump of the LOD Cloud". In: *Proc of ISWC 2017*.