A Formal Theory for Modular ERDF Ontologies

Anastasia Analyti[1], Grigoris Antoniou[1,2], and
Carlos Viegas Damásio[3]

[1] Institute of Computer Science, FORTH-ICS, Greece
[2] Department of Computer Science, University of Crete, Greece
[3] CENTRIA, Departamento de Informatica, Faculdade de Ciencias e Tecnologia,
Universidade Nova de Lisboa, 2829-516 Caparica, Portugal
`{analyti, antoniou}@ics.forth.gr, cd@di.fct.unl.pt`

Abstract. The success of the Semantic Web is impossible without any form of modularity, encapsulation, and access control. In an earlier paper, we extended RDF graphs with weak and strong negation, as well as derivation rules. The ERDF #n-stable model semantics of the extended RDF framework (*ERDF*) is defined, extending RDF(S) semantics. In this paper, we propose a framework for modular ERDF ontologies, called *modular ERDF framework*, which enables collaborative reasoning over a set of ERDF ontologies, while support for hidden knowledge is also provided. In particular, the *modular ERDF stable model semantics* of modular ERDF ontologies is defined, extending the ERDF #n-stable model semantics. Our proposed framework supports local semantics and different points of view, local closed-world and open-world assumptions, and scoped negation-as-failure. Several complexity results are provided.

1 Introduction

Ontologies and automated reasoning are the building blocks of the Semantic Web initiative. Derivation rules can be included in an ontology to define derived concepts based on base concepts. For example, rules allow to define the extension of a class or property based on a complex relation between the extensions of the same or other classes and properties. On the other hand, the inclusion of negative information both in the form of negation-as-failure and explicit negative information is also needed to enable various forms of reasoning. In [1], the Semantic Web language RDFS [8, 6] is extended to accommodate the two negations of Partial Logic [7], namely *weak negation* $\sim$ (expressing negation-as-failure or non-truth) and *strong negation* $\neg$ (expressing explicit negative information or falsity), as well as derivation rules. The new language is called *Extended RDF* (*ERDF*). Specifically, in [1], the *stable model semantics* of ERDF ontologies is developed, based on Partial Logic, extending the model-theoretic semantics of RDFS [6]. The concrete syntax of ERDF is presented in [14].

ERDF enables the combination of closed-world (non-monotonic) and open-world (monotonic) reasoning, in the same framework, through the presence of weak negation (in the body of the program rules) and the new metaclasses *erdf*:*TotalProperty* and *erdf*:*TotalClass*, respectively. In particular, relating strong and weak negation at the interpretation level, ERDF distinguishes two categories of properties and classes [1]. *Partial properties* are properties p that may have truth-value gaps, that is $p(x, y)$ is possibly neither true nor false. *Total properties* are properties p that satisfy *totalness*, that is $p(x, y)$ is either true or false. Partial and total classes c are defined similarly, by replacing $p(x, y)$ by $rdf{:}type(x, c)$. In [1], it is shown that on total properties and total classes, the *Open-World Assumption (OWA)* applies.

ERDF also distinguishes properties and classes that are completely represented in a knowledge base with respect to an (optional) ERDF formula F, corresponding to the *context* where the completion takes place. Such a completeness assumption for *closing* a partial property p by default may be expressed in ERDF by means of the rule $\neg p(?x, ?y) \leftarrow F \wedge {\sim} p(?x, ?y)$ and for a partial class c, by means of the rule $\neg rdf{:}type(?x, c) \leftarrow F \wedge {\sim} rdf{:}type(?x, c)$, where F is an ERDF formula.

Intuitively, an ERDF ontology is the combination of (i) an ERDF graph G containing (implicitly existentially quantified) positive and negative information, and (ii) an ERDF program P containing derivation rules, with possibly all connectives $\sim$, $\neg$, $\supset$, $\wedge$, $\vee$, $\forall$, $\exists$ in the body of a rule, and strong negation $\neg$ in the head of a rule.

In [1], it is shown that stable model entailment conservatively extends RDFS entailment from RDF graphs to ERDF ontologies. Unfortunately, satisfiability and entailment under the ERDF stable model semantics are in general undecidable. This is due to the fact that the RDF vocabulary is infinite. Therefore, to achieve decidability of reasoning in the general case, in [2], we propose a modified semantics, called *ERDF #n-stable model semantics* (for $n \in I\!N$), in which from the RDF vocabulary, we remove the infinite set of terms $\{rdf{:}\_i \mid i > n\}$. The new semantics also extends RDFS entailment from RDF graphs to ERDF ontologies. Additionally, in [2], we provide an equivalence statement between ERDF stable model entailment and ERDF $\#n$-stable model entailment on an ERDF ontology O, in the case that the bodies of the rules in O contain only the connectives $\neg$ and $\wedge$.

The success of the Semantic Web is impossible without any form of modularity, encapsulation, and access control. In this paper, we propose a framework for modular ERDF ontologies, called *modular ERDF framework*, in which a modular ERDF ontology $\mathcal{R}$ is a set of `r`-ERDF ontologies. Intuitively, an `r`-ERDF ontology $O \in \mathcal{R}$ is an ERDF ontology that can import or just reference knowledge about a property or class x from other `r`-ERDF ontologies in $\mathcal{R}$ that define x and are willing to export this knowledge to O. Thus, our modular ERDF framework enables collaborative reasoning over a set of `r`-ERDF ontologies, while support for hidden knowledge is also provided. Additionally, it supports local semantics and different points of view, local closed-world and open-world assumptions, and scoped negation-as-failure.

Specifically, in this paper, we define the *modular* (*ERDF*) *stable models* of an `r`-ERDF ontology w.r.t. a modular ERDF ontology. Several properties of the modular stable model semantics are provided, including that modular stable model entailment extends $\#n$-stable model entailment on ERDF ontologies, and thus also, RDFS entailment on RDF graphs. We show that if $\mathcal{R}$ is a simple modular ERDF ontology (i.e., the bodies of the rules of the `r`-ERDF ontologies in $\mathcal{R}$ contain only the connectives $\sim$, $\neg$, $\wedge$) then query answering under the modular ERDF stable model semantics reduces to query answering under the answer set semantics [5]. Moreover, we provide complexity results for the modular ERDF stable model semantics on (i) simple modular ERDF ontologies, (ii) modular ERDF ontologies without quantifiers, and (ii) general modular ERDF ontologies.

We would like to mention that the goal of our modular ERDF framework is on interconnecting independently developed `r`-ERDF ontologies over the web and *not* on querying a large ontology by decomposing it into smaller sub-ontologies. The latter problem has been considered for answer set semantics in [10], but [10] prohibits the existence of positive recursion among modules, a serious limitation for the Semantic Web setting. In contrast, in our framework, considered `r`-ERDF ontologies may be interconnected via cyclic references. For example, an `r`-ERDF ontology O may be created any time after the independent creation of the `r`-ERDF ontologies on which it depends (which later may be updated, possibly referring to O).

The rest of the paper is organized as follows: In Section 2, we review ERDF graphs, which we extend to `r`-ERDF formulas. Then, we define `r`-ERDF ontologies and valid modular ERDF ontologies. Section 3 defines the modular ERDF interpretations of an `r`-ERDF ontology w.r.t. a modular ERDF ontology. Then, it defines satisfiability of an `r`-ERDF formula by such a modular ERDF interpretation and an `r`-ERDF ontology. In Section 4, we define the modular stable semantics of an `r`-ERDF ontology w.r.t. a modular ERDF ontology, and provide its properties. Further, we provide several complexity results for the modular ERDF stable model semantics. Section 5 reviews related work and concludes the paper.

2 Modular ERDF Ontologies

In this Section, we define `r`-ERDF formulas, valid `r`-ERDF ontologies, and valid modular ERDF ontologies. Additionally, we provide a comprehensive example of a modular ERDF ontology.

A (Web) *vocabulary* V is a set of URI references and/or literals (plain or typed) [6]. We denote the set of all URI references by *URI*, the set of all plain literals by $\mathcal{PL}$, the set of all typed literals by $\mathcal{TL}$, and the set of all literals by $\mathcal{LIT}$. We consider a set *Var* of variable symbols, such that the sets *Var*, *URI*, $\mathcal{LIT}$ are pairwise disjoint. In our examples, variable symbols are prefixed by "?".

Below, we review the definition of an ERDF triple from [1]. Let V be a vocabulary. A (*normal*) *ERDF triple* over V is an expression of the form $p(s, o)$ or $\neg p(s, o)$, where $s, o \in V \cup \mathit{Var}$ are called *subject* and *object*, respectively, and $p \in V \cap \mathit{URI}$ is called *property*.

Below we extend the definition of an ERDF formula, provided in [1], to `r`-ERDF formulas. We consider the connectives $\{\sim, \neg, \wedge, \vee, \supset, \exists, \forall\}$, where $\neg$,

$\sim$, and $\supset$ are called *strong negation*, *weak negation*, and *material implication* respectively. Let V be a vocabulary and let $O_{\texttt{nam}} \subseteq \mathit{URI}$ be a set of $\texttt{r}$-ERDF ontology names. We define $L(V)$ to be the smallest set that contains the ERDF triples over V and is closed with respect to the following conditions: if $F, G \in L(V)$ then $\{\sim F,\ F \wedge G,\ F \vee G,\ F \supset G,\ \exists x F,\ \forall x F\} \subseteq L(V)$, where $x \in \mathit{Var}$. A (*normal*) *ERDF formula* over V is an element of $L(V)$. A *qualified ERDF formula* over V and $O_{\texttt{nam}}$ has the form $F@\mathit{oname}$, where $F \in L(V)$ and $\mathit{oname} \in O_{\texttt{nam}}$ (i.e., F will be evaluated at the $\texttt{r}$-ERDF ontology identified by *oname*).

Definition 1 (r-ERDF formula). Let V be a vocabulary and let $O_{\texttt{nam}} \subseteq \mathit{URI}$. We define $L(V, O_{\texttt{nam}})$ to be the smallest set that (i) contains the ERDF formulas over V and the qualified ERDF formulas over V and $O_{\texttt{nam}}$, and (ii) is closed with respect to the following conditions: if $F, G \in L(V, O_{\texttt{nam}})$ then $\{\sim F,\ F \wedge G,\ F \vee G,\ F \supset G,\ \exists x F,\ \forall x F\} \subseteq L(V, O_{\texttt{nam}})$, where $x \in \mathit{Var}$. An $\texttt{r}$-*ERDF formula* F over V and $O_{\texttt{nam}}$ is an element of $L(V, O_{\texttt{nam}})$. We denote the set of variables appearing in F by $\mathit{Var}(F)$, and the set of free variables[4] appearing in F by $\mathit{FVar}(F)$. $\square$

Next, we review the definition of an ERDF graph G and the skolemization of G from [1]. An *ERDF graph* G over a vocabulary V is a set of ERDF triples over V. We denote the variables appearing in G by $\mathit{Var}(G)$, and the set of URI references and literals appearing in G by V_G. Intuitively, an ERDF graph G represents an existentially quantified conjunction of ERDF triples. Specifically, let $G = \{t_1, ..., t_m\}$ be an ERDF graph, and let $\mathit{Var}(G) = \{x_1, ..., x_k\}$. Then, G represents the ERDF formula $\mathit{formula}(G) = \exists ?x_1, ..., \exists ?x_k\ t_1 \wedge ... \wedge t_m$. Existentially quantified variables in ERDF graphs are handled by *skolemization*. Let G be an ERDF graph. The *skolemization function* of G is an 1:1 mapping $sk_G : \mathit{Var}(G) \rightarrow \mathit{URI}$, where for each $x \in \mathit{Var}(G)$, $sk_G(x)$ is an artificial URI, denoted by $G{:}x$. The *skolemization* of G, denoted by $sk(G)$, is the ground ERDF graph derived from G after replacing each $x \in \mathit{Var}(G)$ by $sk_G(x)$.

Below, we extend the definitions of ERDF rule and ERDF program, provided in [1], to $\texttt{r}$-ERDF rule and $\texttt{r}$-ERDF program, respectively.

Definition 2 (r-ERDF rule, r-ERDF program). An $\texttt{r}$-*ERDF rule* r over a vocabulary V and $O_{\texttt{nam}} \subseteq \mathit{URI}$ is an expression of the form: $G \leftarrow F$, where $F \in L(V, O_{\texttt{nam}}) \cup \{\texttt{true}\}$ (called *condition*) and G (called *conclusion*) is either an ERDF triple over V or $\texttt{false}$. We assume that no bound variable in F appears free in G. We denote the set of variables and the set of free variables of r by $\mathit{Var}(r)$ and $\mathit{FVar}(r)$[5], respectively. Additionally, we write $\mathit{Cond}(r) = F$ and $\mathit{Concl}(r) = G$.
An $\texttt{r}$-*ERDF program* P over a vocabulary V and $O_{\texttt{nam}} \subseteq \mathit{URI}$ is a finite set of $\texttt{r}$-ERDF rules over V and $O_{\texttt{nam}}$. We denote the set of URI references and literals appearing in P by V_P. $\square$

Below, we extend the definition of an ERDF ontology, provided in [1], to an $\texttt{r}$-ERDF ontology.

[4] Without loss of generality, we assume that a variable cannot have both free and bound occurrences in F, and more than one bound occurrence.

[5] $\mathit{FVar}(r) = \mathit{FVar}(F) \cup \mathit{FVar}(G)$.

Definition 3 (r-ERDF ontology). An r-*ERDF ontology* O over a vocabulary V and $O_{\mathtt{nam}} \subseteq \mathit{URI}$ is a triple $O = \langle \mathit{Nam}_O, L_O, \mathit{Int}_O \rangle$, where: (i) $\mathit{Nam}_O \in O_{\mathtt{nam}}$ is the *name* of O, (ii) $L_O = \langle G_O, P_O, \rangle$, is the *logic* of O, where G_O is an ERDF graph over V and P_O is an r-ERDF program over V and $O_{\mathtt{nam}}$, and (iii) $\mathit{Int}_O = \langle \mathit{Exp}^{\mathtt{pr}}_O, \mathit{Exp}^{\mathtt{cl}}_O, \mathit{Imp}^{\mathtt{pr}}_O, \mathit{Imp}^{\mathtt{cl}}_O \rangle$ is the *interface* of O, where: For $\mathtt{t} \in \{\mathtt{pr}, \mathtt{cl}\}$, it holds that:

- $\mathit{Exp}^{\mathtt{t}}_O$ is a set of pairs $\langle x, \mathit{Exp} \rangle$, where $x \in V$ and $\mathit{Exp} \subseteq O_{\mathtt{nam}} - \{\mathit{Nam}_O\}$ or $\mathit{Exp} = \{*\}$. It holds that if $\langle x, \mathit{Exp} \rangle$ and $\langle x, \mathit{Exp}' \rangle \in \mathit{Exp}^{\mathtt{t}}_O$ then $\mathit{Exp} = \mathit{Exp}'$.
  We define: $\mathit{Exported}^{\mathtt{t}}_O = \{x \mid \exists \langle x, \mathit{Exp} \rangle \in \mathit{Exp}^{\mathtt{t}}_O\}$ and $\mathit{Export}^{\mathtt{t}}_O(x) = \mathit{Exp}$.
- $\mathit{Imp}^{\mathtt{t}}_O$ is a set of pairs $\langle x, \mathit{Imp} \rangle$, where $x \in V$, and $\mathit{Imp} \subseteq O_{\mathtt{nam}} - \{\mathit{Nam}_O\}$ or $\mathit{Imp} = \{*\}$. It holds that if $\langle x, \mathit{Imp} \rangle$ and $\langle x, \mathit{Imp}' \rangle \in \mathit{Imp}^{\mathtt{t}}_O$ then $\mathit{Imp} = \mathit{Imp}'$.
  We define: $\mathit{Imported}^{\mathtt{t}}_O = \{x \mid \exists \langle x, \mathit{Imp} \rangle \in \mathit{Imp}^{\mathtt{t}}_O\}$ and $\mathit{Import}^{\mathtt{t}}_O(x) = \mathit{Imp}$.

Let O be an r-ERDF ontology. Intuitively, each pair $\langle x, \mathit{Exp} \rangle \in \mathit{Exp}^{\mathtt{pr}}_O$ (resp. $\langle x, \mathit{Exp} \rangle \in \mathit{Exp}^{\mathtt{cl}}_O$) corresponds to an `export` declaration of O, where x is a property (resp. class) exported by O and Exp is the list of r-ERDF ontologies to which O is willing to export x. If O is willing to export x to any requesting r-ERDF ontology then $\mathit{Exp} = \{*\}$.

Similarly, each pair $\langle x, \mathit{Imp} \rangle \in \mathit{Imp}^{\mathtt{pr}}_O$ (resp. $\langle x, \mathit{Imp} \rangle \in \mathit{Imp}^{\mathtt{cl}}_O$) corresponds to an `import` declaration of O, where x is a property (resp. class) requested by O, and Imp is the list of r-ERDF ontologies from which x is requested. If O requests x from any providing r-ERDF ontology then $\mathit{Imp} = \{*\}$. Obviously, we do not allow duplicate `export` and `import` declarations for classes and properties in O.

Definition 4 (Modular ERDF ontology). A *modular ERDF ontology* (*MEO*) $\mathcal{R}$ is a set of r-ERDF ontologies. □

Example 1. Consider the modular ERDF ontology $\mathcal{R} = \{O_1, O_2, O_3, O_4, O_5\}$, shown in Figure 1[6]. Ontology O_1, with Nam_{O_1} =`<http://geography.int>`, provides geographical information, stating that the list of European countries is positively closed (w.r.t. the list of countries). This local CWA is expressed by the single rule in P_{O_1}. Ontology O_2, with Nam_{O_2} =`<http://europa.eu>`, defines the list of European Union countries (which does not include `Croatia`) and states that this list is open (w.r.t. the resources of O_2)[7] by declaring the class `eu:CountryEU` as total. This local OWA is expressed by the first ERDF triple in G_{O_2}. Ontology O_3, with Nam_{O_3} =`<http://www.pyramis.gr>`, provides information regarding the package tours of the greek travel agency *Pyramis*. Similarly, ontology O_4, with Nam_{O_4} =`<http://www.travel_plan.gr>`, provides information regarding the package tours of the greek travel agency *Travel Plan*.

Finally, ontology O_5, with Nam_{O_5} =`<http://www.anne_travel_pref.gr>`, presents the travel preferences of Anne. Specifically, Anne prefers either (i) a trip to a non-European country by *Pyramis* that visits only one city, or (ii) a trip to an EU country by *Travel Plan* that visits at least one city, or (iii) a trip to a European but not EU country by *Travel Plan* that visits the capital of the

[6] Following usual convention, we have replaced $\wedge$ by "," in the program rules.
[7] Note that ontology O_2 imports class `geo:Country` from ontology O_1.

Ontology O_1

```
⟨http://geography.int⟩

exports class geo:Country to ∗.
exports class geo:Europ_Country to ∗.
exports property geo:capital to ∗.

G_O1 =
rdfs:subclass(geo:Europ_Country,
              geo:Country).
rdf:type(geo:Egypt,geo:Country).
rdf:type(geo:Italy,geo:Europ_Country).
rdf:type(geo:Croatia,geo:Europ_Country).
geo:capital(geo:Cairo,geo:Egypt).
geo:capital(geo:Zagreb,geo:Croatia).
...

P_O1 =
¬ rdf:type(?x,geo:Europ_Country) ←
         rdf:type(?x,geo:Country),
       ∼ rdf:type(?x,geo:Europ_Country).
```

Ontology O_2

```
⟨http://europa.eu⟩

imports class geo:Country from
          ⟨http://geography.int⟩.
exports class eu:CountryEU to ∗.

G_O2 =
rdf:type(eu:CountryEU, erdf:TotalClass).
rdf:type(geo:Italy,eu:CountryEU).
rdf:type(geo:Greece,eu:CountryEU).
...
```

Ontology O_3

```
⟨http://www.pyramis.gr⟩

exports property vac:travel to ∗.
exports property vac:visit to ∗.

G_O3 =
vac:travel(pyr:package1,geo:Egypt).
vac:visit(pyr:package1,geo:Cairo).
vac:travel(pyr:package2,geo:Egypt).
vac:visit(pyr:package2,geo:Cairo).
vac:visit(pyr:package2,geo:Luxor).
```

Ontology O_4

```
⟨http://www.travel_plan.gr⟩

exports property vac:travel to ∗.
exports property vac:visit to ∗.

G_O4 =
vac:travel(trav:package1,geo:Italy).
vac:visit(trav:package1,geo:Rome).
vac:travel(trav:package2,geo:Croatia).
vac:visit(trav:package2,geo:Zagreb).
vac:visit(trav:package2,geo:Trogir).
```

Ontology O_5

```
⟨http://www.anne_travel_pref.gr⟩

imports class geo:Europ_Country from ⟨http://geography.int⟩.
imports property geo:capital from ⟨http://geography.int⟩.
imports class eu:CountryEU from ⟨http://europa.eu⟩.
imports property vac:travel from ∗.
imports property vac:visit from ∗.
exports property ann:choose_trav_package to ⟨http://www.peter_travel_pref.gr⟩.

P_O5 =
eq:id(?x,?x) ← true.

ann:choose_trav_package(?package,?country) ← ¬ rdf:type(?country,geo:Europ_Country),
  (vac:travel(?package,?country), vac:visit(?package,?city))@⟨http://www.pyramis.gr⟩,
  ∀ ?city′ vac:visit(?package,?city′)@⟨http://www.pyramis.gr⟩ ⊃ eq:id(?city,?city′).

ann:choose_trav_package(?package,?country) ← rdf:type(?country,geo:CountryEU),
  (vac:travel(?package,?country), vac:visit(?package,?city),
   vac:visit(?package,?city′))@⟨http://www.travel_plan.gr⟩, ∼ eq:id(?city,?city′).

ann:choose_trav_package(?package,?country) ← rdf:type(?country,geo:Europ_Country),
 ¬ rdf:type(?country,geo:CountryEU),
   (vac:travel(?package,?country), vac:visit(?package,?city))@⟨http://www.travel_plan.gr⟩,
   geo:capital(?city,?country).
```

Fig. 1. A modular ERDF ontology

country. Note that O_5 imports the properties `vac:travel` and `vac:visit` from any providing `r`-ERDF ontology in $\mathcal{R}$ (that is, O_3 and O_4). Additionally, note that `r`-ERDF ontology O_5 exports property `ann:choose_trav_package` to an `r`-ERDF ontology, named `<http://www.peter_travel_pref.gr>`, not in $\mathcal{R}$. $\Box$

Let $\mathcal{R}$ be a modular ERDF ontology, let $O \in \mathcal{R}$, and let $x \in \mathit{Exported}^{\mathtt{t}}_{O}$, for $\mathtt{t} \in \{\mathtt{pr}, \mathtt{cl}\}$. We define:

$$\mathit{Export}^{\mathtt{t}}_{O,\mathcal{R}}(x) = \begin{cases} \{\mathit{Nam}_{O'} \mid O' \in \mathcal{R} - \{O\}\} & \text{if } \mathit{Export}^{\mathtt{t}}_{O}(x) = \{*\} \\ \mathit{Export}^{\mathtt{t}}_{O}(x) \cap \{\mathit{Nam}_{O'} \mid O' \in \mathcal{R}\} & \text{otherwise} \end{cases}$$

Intuitively, $\mathit{Export}^{\mathtt{pr}}_{O,\mathcal{R}}(x)$ (resp. $\mathit{Export}^{\mathtt{cl}}_{O,\mathcal{R}}(x)$) denotes the `r`-ERDF ontologies in $\mathcal{R}$ to which O is willing to export property (resp. class) x.

Example 2. Consider the modular ERDF ontology $\mathcal{R}$ of Example 1. Then, it holds that: $\mathit{Export}^{\mathtt{cl}}_{O_2,\mathcal{R}}(\mathtt{eu:CountryEU}) = \{O_1, O_3, O_4, O_5\}$, because $\mathit{Export}^{\mathtt{cl}}_{O_2}(\mathtt{eu:CountryEU}) = \{*\}$. Additionally, it holds that $\mathit{Export}^{\mathtt{pr}}_{O_5,\mathcal{R}}(\mathtt{ann:choose\_trav\_package})=\{\}$. $\Box$

Let $\mathcal{R}$ be a modular ERDF ontology, let $O \in \mathcal{R}$, and let $x \in \mathit{Imported}^{\mathtt{t}}_{O}$, for $\mathtt{t} \in \{\mathtt{pr}, \mathtt{cl}\}$. We define:

$$\mathit{Import}^{\mathtt{t}}_{O,\mathcal{R}}(x) = \begin{cases} \mathit{ExportingTo}^{\mathtt{t}}_{\mathcal{R}}(x, O) & \text{if } \mathit{Import}^{\mathtt{t}}_{O}(x) = \{*\} \\ \mathit{Import}^{\mathtt{t}}_{O}(x) \cap \mathit{ExportingTo}^{\mathtt{t}}_{\mathcal{R}}(x, O) & \text{otherwise,} \end{cases}$$

where $\mathit{ExportingTo}^{\mathtt{t}}_{\mathcal{R}}(x, O) = \{\mathit{Nam}_{O'} \mid O' \in \mathcal{R},\ \mathit{Nam}_O \in \mathit{Export}^{\mathtt{t}}_{O',\mathcal{R}}(x)\}$. Intuitively, $\mathit{ExportingTo}^{\mathtt{pr}}_{\mathcal{R}}(x, O)$ (resp. $\mathit{ExportingTo}^{\mathtt{cl}}_{\mathcal{R}}(x, O)$) denotes the `r`-ERDF ontologies in $\mathcal{R}$ that are willing to export property (resp. class) x to O. Additionally, $\mathit{Import}^{\mathtt{pr}}_{O,\mathcal{R}}(x)$ (resp. $\mathit{Import}^{\mathtt{cl}}_{O,\mathcal{R}}(x)$) denotes the `r`-ERDF ontologies in $\mathcal{R}$ from which O imports property (resp. class) x.

Example 3. For the modular ERDF ontology $\mathcal{R}$ of Example 1, $\mathit{ExportingTo}^{\mathtt{pr}}_{\mathcal{R}}(\mathtt{vac:travel}, O_5) = \{O_3, O_4\}$. Additionally, $\mathit{Import}^{\mathtt{pr}}_{O_5,\mathcal{R}}(\mathtt{vac:travel}) = \{O_3, O_4\}$. $\Box$

In order for a modular rule base to be *valid*, it has to satisfy a number of validity constraints.

Definition 5 (Valid modular ERDF ontology). A modular ERDF ontology $\mathcal{R}$ is *valid* iff:

1. If $O, O' \in \mathcal{R}$ and $O \neq O'$ then $\mathit{Nam}_O \neq \mathit{Nam}_{O'}$.
2. If $O \in \mathcal{R}$ and $x \in \mathit{Imported}^{\mathtt{t}}_{O}$, for $\mathtt{t} \in \{\mathtt{pr}, \mathtt{cl}\}$, then $\mathit{Import}^{\mathtt{t}}_{O}(x) = \{*\}$ or $\mathit{Import}^{\mathtt{t}}_{O}(x) \subseteq \mathit{ExportingTo}^{\mathtt{t}}_{\mathcal{R}}(x, O)$.
3. If $O \in \mathcal{R}$ and $r \in P_O$ such that a qualified ERDF formula $F@\mathit{Nam}_{O'}$ appears in $\mathit{Cond}(r)$ then: (i) $O' \in \mathcal{R}$, (ii) for each $p(s, o)$, where $p \neq \mathit{rdf{:}type}$, appearing in F, it holds that $O \in \mathit{Export}^{\mathtt{pr}}_{O',\mathcal{R}}(p)$, and (iii) for each $\mathit{rdf{:}type}(x, c)$, appearing in F, it holds that (a) $O \in \mathit{Export}^{\mathtt{pr}}_{O',\mathcal{R}}(\mathit{rdf{:}type})$ or (b) $c \in \mathit{URI}$ and $O \in \mathit{Export}^{\mathtt{cl}}_{O',\mathcal{R}}(c)$. $\Box$

Let $\mathcal{R}$ be a valid modular ERDF ontology. Constraint (1) of Definition 5 expresses that different $\mathtt{r}$-ERDF ontologies in $\mathcal{R}$ should have different names in order to be uniquely identified. Let $O \in \mathcal{R}$. Constraint (2) expresses that if O requests a property or class x *explicitly* from an $\mathtt{r}$-ERDF ontology O' then it should hold that $O' \in \mathcal{R}$ and O' is willing to export x to O. Assume now that it exists $r \in P_O$ s.t. $Cond(r)$ refers to an ERDF formula F of an $\mathtt{r}$-ERDF ontology O'. Constraint (3.i) expresses that it should hold $O' \in \mathcal{R}$. Constraint (3.ii) expresses that if r refers to $p(s,o)$ of O', where $p \neq rdf{:}type$, then O' should be willing to export property p to O. Additionally, constraint (3.iii) expresses that if O refers to $rdf{:}type(x,c)$ of O' then O' should be willing to either (a) export to O the property $rdf{:}type$, expressing that all classes of O' are exported to O, or (b) export to O just the class c (if $c \in URI$).

Example 4. Modular rule base $\mathcal{R}$ of Example 1 is valid. $\square$

In this work, we consider valid modular ERDF ontologies, only. Additionally, by $\mathcal{R}$, we will denote a valid modular ERDF ontology.

3 Modular ERDF and Herbrand Interpretations

In this section, we define the modular ERDF interpretations of an $\mathtt{r}$-ERDF ontology w.r.t. a modular ERDF ontology. Additionally, we define satisfaction of an $\mathtt{r}$-ERDF formula by such a modular ERDF interpretation and an $\mathtt{r}$-ERDF ontology. Further, we define the modular Herbrand interpretations of an $\mathtt{r}$-ERDF ontology w.r.t. a modular ERDF ontology.

Below we review the definition of a partial interpretation of a vocabulary V [1], which is an extension of the definition of a simple interpretation of V [6], such that each property is associated not only with a truth extension but also with a falsity extension, allowing for partial properties.

Definition 6 (Partial interpretation of a vocabulary). A *partial interpretation* I of a vocabulary V_I consists of:

- A non-empty set of resources Res_I, called the *domain* or *universe* of I.
- A set of properties $Prop_I$.
- A vocabulary interpretation mapping $I_{\mathtt{V}} : V_I \cap URI \rightarrow Res_I \cup Prop_I$.
- A property-truth extension mapping $PT_I : Prop_I \rightarrow \mathcal{P}(Res_I \times Res_I)$.
- A property-falsity extension mapping $PF_I : Prop_I \rightarrow \mathcal{P}(Res_I \times Res_I)$.
- A mapping $IL_I : V_I \cap \mathcal{TL} \rightarrow Res_I$.
- A set of literal values $LV_I \subseteq Res_I$, which contains $V \cap \mathcal{PL}$.

We define the mapping: $I : V_I \rightarrow Res_I \cup Prop_I$ such that: (i) $I(x) = I_{\mathtt{V}}(x)$, $\forall x \in V_I \cap URI$, (ii) $I(x) = x$, $\forall\ x \in V_I \cap \mathcal{PL}$, and (iii) $I(x) = IL_I(x)$, $\forall\ x \in V_I \cap \mathcal{TL}$. $\square$

A partial interpretation I is *coherent* iff for all $x \in Prop_I$, $PT_I(x) \cap PF_I(x) = \emptyset$.

Let $O \in \mathcal{R}$. Below we define the dependencies of O w.r.t. $\mathcal{R}$.

Definition 7 (Dependencies of an r-ERDF ontology w.r.t. a MEO). Let $O \in \mathcal{R}$. The *dependencies* of O w.r.t. $\mathcal{R}$, denoted by $D_O^{\mathcal{R}}$, is the minimum set of r-ERDF ontologies s.t.: (i) $O \in D_O^{\mathcal{R}}$, (ii) if $O' \in D_O^{\mathcal{R}}$ and it exists $x \in \mathit{Imported}_{O'}^{\mathtt{t}}$, for $\mathtt{t} \in \{\mathtt{pr}, \mathtt{cl}\}$, s.t. $\mathit{Nam}_{O''} \in \mathit{Import}_{O',\mathcal{R}}^{\mathtt{t}}(x)$ then $O'' \in D_O^{\mathcal{R}}$, and (iii) if $O' \in D_O^{\mathcal{R}}$, $r \in P_{O'}$, and it exists a qualified ERDF formula $F@\mathit{Nam}_{O''}$ in $\mathit{Cond}(r)$ then $O'' \in D_O^{\mathcal{R}}$. $\square$

Example 5. Consider the modular ERDF ontology $\mathcal{R}$ of Example 1. It holds: $D_{O_1}^{\mathcal{R}} = \{O_1\}$, $D_{O_2}^{\mathcal{R}} = \{O_2, O_1\}$, and $D_{O_5}^{\mathcal{R}} = \{O_5, O_1, O_2, O_3, O_4\}$. $\square$

The vocabulary of RDF, $\mathcal{V}_{RDF}$, is a set of *URI* references in the *rdf*: namespace [6], and the vocabulary of RDFS, $\mathcal{V}_{RDFS}$, is a set of *URI* references in the *rdfs*: namespace [6]. Let $n \in I\!N$. We define $\mathcal{V}_{RDF}^{\#n} = \mathcal{V}_{RDF} - \{\mathit{rdf}{:}\_i \mid i > n\}$. The *vocabulary of ERDF* is defined as $\mathcal{V}_{ERDF} = \{\mathit{erdf}{:}\mathit{TotalClass}, \mathit{erdf}{:}\mathit{TotalProperty}\}$.

Let $O \in \mathcal{R}$. We define: (i) $n_O = 0$, if $(V_{G_O} \cup V_{P_O}) \cap \{\mathit{rdf}{:}\_i \mid i \geq 1\} = \emptyset$, and (ii) $n_O = \mathit{max}(\{i \in I\!N \mid \mathit{rdf}{:}\_i \in V_{G_O} \cup V_{P_O}\})$, otherwise. Further, we define: $n_{\mathcal{R}} = \mathit{max}(\{n_O \mid O \in \mathcal{R}\} \cup \{1\})$. Intuitively, $n_{\mathcal{R}}$ is the largest i ($i \in I\!N$) such that $\mathit{rdf}{:}\_i$ appears in an $O \in \mathcal{R}$. In the case that no such an $\mathit{rdf}{:}\_i$ exists then $n_{\mathcal{R}} = 1$. Recall that the $\mathit{rdf}{:}\_i$ properties are used in RDF(S) [6] to express members of containers (i.e. bags, sequences, and alternatives), which are in practice finitely limited.

Let $O \in \mathcal{R}$, and let $n \in I\!N$. The *n#-vocabulary* of O is defined as: $V_O^{\#n} = V_{sk(G_O)} \cup V_{P_O} \cup \mathit{Exported}_O^{\mathtt{pr}} \cup \mathit{Exported}_O^{\mathtt{cl}} \cup \mathit{Imported}_O^{\mathtt{pr}} \cup \mathit{Imported}_O^{\mathtt{cl}} \cup \mathcal{V}_{RDF}^{\#n} \cup \mathcal{V}_{RDFS} \cup \mathcal{V}_{ERDF}$. The *vocabulary* of O w.r.t. $\mathcal{R}$ is defined as: $V_{O,\mathcal{R}} = \cup\{V_{O'}^{\#n_{\mathcal{R}}} \mid O' \in D_O^{\mathcal{R}}\}$. Intuitively, $V_{O,\mathcal{R}}$ corresponds to the local domain of O w.r.t. R.

Let $n \in I\!N$. Below we define the modular ERDF interpretations of an r-ERDF ontology w.r.t. a modular ERDF ontology. In this definition, we use the definition of an *ERDF #n-interpretation* over a vocabulary V (see [2]), not reviewed here due to space limitations. Intuitively, an *ERDF #n-interpretation* I of a vocabulary V is a partial interpretation of $V_I = V \cup \mathcal{V}_{RDF}^{\#n} \cup \mathcal{V}_{RDFS} \cup \mathcal{V}_{ERDF}$ that assigns truth and falsity extensions to the classes[8] and properties in V_I, satisfying: (i) all semantic conditions of an RDFS interpretation [6] of V, except these referring to $\{\mathit{rdf}{:}\_i \mid i > n\}$ terms, as well as (ii) new semantic conditions, particular to ERDF.

Definition 8 (Modular ERDF interpretation). Let $O \in \mathcal{R}$. A *modular ERDF interpretation* of O w.r.t. $\mathcal{R}$ is a set $\mathsf{I} = \{I_{O'} \mid O' \in D_O^{\mathcal{R}}\}$, where $I_{O'}$ is an ERDF $\#n_{\mathcal{R}}$-interpretation of $V_{O',\mathcal{R}}$ and it holds:

1. If $O' \in D_O^{\mathcal{R}}$, $p \in \mathit{Imported}_{O',\mathcal{R}}^{\mathtt{pr}}$, and $\mathit{Nam}_{O''} \in \mathit{Import}_{O',\mathcal{R}}^{\mathtt{pr}}(p)$ then $PT_{I_{O'}}(I_{O'}(p)) \supseteq PT_{I_{O''}}(I_{O''}(p))$, and $PF_{I_{O'}}(I_{O'}(p)) \supseteq PF_{I_{O''}}(I_{O''}(p))$, and

[8] The *truth* and *falsity extension* of a class $c \in V_I$ is indicated by $CT_I(I(c))$ and $CF_I(I(c))$, respectively. It holds: (i) $x \in CT_I(y)$ iff $\langle x, y\rangle \in PT_I(I(\mathit{rdf}{:}\mathit{type}))$, and (ii) $x \in CF_I(y)$ iff $\langle x, y\rangle \in PF_I(I(\mathit{rdf}{:}\mathit{type}))$.

2. If $O' \in D_O^{\mathcal{R}}$, $c \in \mathit{Imported}^{\mathtt{cl}}_{O',\mathcal{R}}$, and $\mathit{Nam}_{O''} \in \mathit{Import}^{\mathtt{cl}}_{O',\mathcal{R}}(c)$ then $CT_{I_{O'}}(I_{O'}(c)) \supseteq CT_{I_{O''}}(I_{O''}(c))$, and $CF_{I_{O'}}(I_{O'}(c)) \supseteq CF_{I_{O''}}(I_{O''}(c))$. □

Below, we define satisfaction of an r-ERDF formula w.r.t. a modular ERDF interpretation, an r-ERDF ontology, and a valuation. First, we provide an auxiliary definition. Let I be a partial interpretation of a vocabulary V_I, let Res be a set, and let v be a partial function $v : \mathit{Var} \to \mathit{Res}$ (called *valuation*). If $x \in \mathit{Var}$, we define $[I + v](x) = v(x)$. If $x \in V_I$, we define $[I + v](x) = I(x)$.

Definition 9. (Satisfaction of an r-ERDF formula w.r.t. a modular ERDF interpretation, an r-ERDF ontology, and a valuation) Let $O \in \mathcal{R}$. Let $\mathsf{I} = \{I_{O'} \mid O' \in D_O^{\mathcal{R}}\}$ be a modular ERDF interpretation of O w.r.t. $\mathcal{R}$. Additionally, let F, G be r-ERDF formulas over $\{\mathit{Nam}_{O'} \mid O' \in D_O^{\mathcal{R}}\}$. For each $O', O'' \in D_O^{\mathcal{R}}$ and for each mapping $v : \mathit{Var}(F) \to \mathit{Res}_{I_{O'}}$:

– If $F = p(s,o)$ then $\langle \mathsf{I}, O', v\rangle \models F$ iff $p \in V_{I_{O'}} \cap \mathit{URI}$, $s, o \in V_{I_{O'}} \cup \mathit{Var}$, $I_{O'}(p) \in \mathit{Prop}_{I_{O'}}$, and $\langle [I_{O'} + v](s), [I_{O'} + v](o)\rangle \in PT_{I_{O'}}(I_{O'}(p))$.
– If $F = \neg p(s,o)$ then $\langle \mathsf{I}, O', v\rangle \models F$ iff $p \in V_{I_{O'}} \cap \mathit{URI}$, $s, o \in V_{I_{O'}} \cup \mathit{Var}$, $I_{O'}(p) \in \mathit{Prop}_{I_{O'}}$, and $\langle [I_{O'} + v](s), [I_{O'} + v](o)\rangle \in PF_{I_{O'}}(I_{O'}(p))$.
– If $F = {\sim}G$ then $\langle \mathsf{I}, O', v\rangle \models F$ iff $V_G \subseteq V_{I_{O'}}$ and $\langle \mathsf{I}, O', v\rangle \not\models G$.
– If $F = F_1 \wedge F_2$ then $\langle \mathsf{I}, O', v\rangle \models F$ iff $\langle \mathsf{I}, O', v\rangle \models F_1$ and $\langle \mathsf{I}, O', v\rangle \models F_2$.
– If $F = F_1 \vee F_2$ then $\langle \mathsf{I}, O', v\rangle \models F$ iff $\langle \mathsf{I}, O', v\rangle \models F_1$ or $\langle I, O', v\rangle \models F_2$.
– If $F = F_1 \supset F_2$ then $\langle \mathsf{I}, O', v\rangle \models F$ iff $\langle \mathsf{I}, O', v\rangle \models {\sim}F_1 \vee F_2$.
– If $F = \exists x\ G$ then $\langle \mathsf{I}, O', v\rangle \models F$ iff there exists a mapping $u : \mathit{Var}(G) \to \mathit{Res}_{I_{O'}}$ such that $u(y) = v(y)$, $\forall y \in \mathit{Var}(G) - \{x\}$, and $\langle \mathsf{I}, O', u\rangle \models G$.
– If $F = \forall x\ G$ then $\langle \mathsf{I}, O', v\rangle \models F$ iff for all mappings $u : \mathit{Var}(G) \to \mathit{Res}_{I_{O'}}$ such that $u(y) = v(y)$, $\forall y \in \mathit{Var}(G) - \{x\}$, it holds $\langle \mathsf{I}, O', u\rangle \models G$.
– If $F = G@\mathit{Nam}_{O''}$ then $\langle \mathsf{I}, O', v\rangle \models F$ iff $\langle \mathsf{I}, O'', v\rangle \models G$. □

Let I be a modular ERDF interpretation of O w.r.t. $\mathcal{R}$ and let F be an r-ERDF formula. We define: $\langle \mathsf{I}, O'\rangle \models F$ iff for each mapping $v : \mathit{Var}(F) \to \mathit{Res}_{I_{O'}}$, it holds that $\langle \mathsf{I}, O', v\rangle \models F$. Additionally, let G be an ERDF graph. We define: $\langle \mathsf{I}, O'\rangle \models G$ iff $\langle \mathsf{I}, O'\rangle \models \mathit{formula}(G)$. We assume that for every function $v : \mathit{Var} \to \mathit{Res}_{I_{O'}}$, it holds that $\langle \mathsf{I}, O', v\rangle \models \mathtt{true}$ and $\langle \mathsf{I}, O', v\rangle \not\models \mathtt{false}$.

Below, we define the modular models of an r-ERDF ontology w.r.t. a modular ERDF ontology.

Definition 10 (Modular ERDF model). Let $O \in \mathcal{R}$. Let $\mathsf{I} = \{I_{O'} \mid O' \in D_O^{\mathcal{R}}\}$ be a modular ERDF interpretation of O w.r.t. $\mathcal{R}$ and let $O' \in D_O^{\mathcal{R}}$:

– We say that $\langle \mathsf{I}, O'\rangle$ *satisfies* an r-ERDF rule r, denoted by $\langle \mathsf{I}, O'\rangle \models r$, iff it holds: For all mappings $v : \mathit{Var}(r) \to \mathit{Res}_{I_{O'}}$, if $\langle \mathsf{I}, O', v\rangle \models \mathit{Cond}(r)$ then $\langle \mathsf{I}, O', v\rangle \models \mathit{Concl}(r)$.
– We say that I is a *modular* (*ERDF*) *model* of O w.r.t. $\mathcal{R}$, denoted by $\mathsf{I} \models_{\mathcal{R}} O$, iff for all $O' \in D_O^{\mathcal{R}}$, $\langle \mathsf{I}, O'\rangle \models G_{O'}$ and $\langle \mathsf{I}, O'\rangle \models r$, $\forall\, r \in P_{O'}$. □

Let $O \in \mathcal{R}$. We denote by $\mathit{Res}^{\mathtt{H}}_{O,\mathcal{R}}$ the union of $V_{O,\mathcal{R}}$ and the set of XML values of the well-typed XML literals in $V_{O,\mathcal{R}}$ minus the well-typed XML literals. Below we define the modular Herbrand interpretations of O w.r.t. $\mathcal{R}$, extending the definition of a Herbrand interpretation of an ERDF ontology [1].

Definition 11 (Modular ERDF Herbrand interpretation). Let $O \in \mathcal{R}$. Let $\mathsf{I} = \{I_{O'} | \ O' \in D^{\mathcal{R}}_O\}$ be a modular ERDF interpretation of O w.r.t. $\mathcal{R}$. We say that I is a *modular* (*ERDF*) *Herbrand interpretation* of O w.r.t. $\mathcal{R}$ iff for each $O' \in D^{\mathcal{R}}_O$:

- $Res_{I_{O'}} = Res^{\mathtt{H}}_{O',\mathcal{R}}$.
- $I_{O'_{\mathrm{V}}}(x) = x$, for all $x \in V_{O',\mathcal{R}} \cap URI$.
- $IL_{I_{O'}}(x) = x$, if x is a typed literal in $V_{O',\mathcal{R}}$ other than a well-typed XML literal, and $IL_{I_{O'}}(x)$ is the XML value of x, if x is a well-typed XML literal in $V_{O',\mathcal{R}}$.

We denote the set of modular Herbrand interpretations of O w.r.t. $\mathcal{R}$ by $\mathcal{I}^{\mathtt{H}}_{O,\mathcal{R}}$. $\square$

Let $O \in \mathcal{R}$. Let $\mathsf{I} = \{I_{O'} | \ O' \in D^{\mathcal{R}}_O\}$ be a modular Herbrand interpretation of O w.r.t. $\mathcal{R}$. We say that I is a *modular* (*ERDF*) *Herbrand model* of O w.r.t. $\mathcal{R}$ iff for all $O' \in D^{\mathcal{R}}_O$, (i) $\langle \mathsf{I}, O' \rangle \models sk(G_{O'})$ and (ii) for all $r \in P_{O'}$, $\langle \mathsf{I}, O' \rangle \models r$. We denote the set of modular Herbrand models of O w.r.t. $\mathcal{R}$ by $\mathcal{M}^{\mathtt{H}}_{O,\mathcal{R}}$.

It holds that: if M is a modular Herbrand model of O w.r.t. $\mathcal{R}$ then M is a modular model of O w.r.t. $\mathcal{R}$.

4 Modular Stable Models & Complexity Results

In this Section, we define the modular stable models of an r-ERDF ontology w.r.t. a modular ERDF ontology, and provide some of their properties. Additionally, we provide several complexity results.

Let $O \in \mathcal{R}$. We proceed by defining a partial ordering on the modular Herbrand interpretations of O w.r.t. $\mathcal{R}$.

Definition 12 (Modular Herbrand interpretation ordering). Let $O \in \mathcal{R}$. Let $\mathsf{I}, \mathsf{J} \in \mathcal{I}^{\mathtt{H}}_{O,\mathcal{R}}$. We say that J *extends* I, denoted by $\mathsf{I} \leq \mathsf{J}$ (or $\mathsf{J} \geq \mathsf{I}$) iff: For all $O' \in D^{\mathcal{R}}_O$, it holds that (i) $Prop_{I_{O'}} \subseteq Prop_{J_{O'}}$, and for all $p \in Prop_{I_{O'}}$, it holds $PT_{I_{O'}}(p) \subseteq PT_{J_{O'}}(p)$ and $PF_{I_{O'}}(p) \subseteq PF_{J_{O'}}(p)$. $\square$

Let $O \in \mathcal{R}$. The intuition behind Definition 12 is that by extending a modular Herbrand interpretation of O w.r.t. $\mathcal{R}$, we extend both the truth and falsity extension for all properties of $O' \in D^{\mathcal{R}}_O$, and thus (since *rdf:type* is a property), for all classes.

Let $\mathcal{I} \subseteq \mathcal{I}^{\mathtt{H}}_{O,\mathcal{R}}$. We define $minimal(\mathcal{I}) = \{\mathsf{I} \in \mathcal{I} \mid \ \nexists \mathsf{J} \in \mathcal{I} : \ \mathsf{J} \neq \mathsf{I} \text{ and } \mathsf{J} \leq \mathsf{I}\}$. Let $\mathsf{I}, \mathsf{J} \in \mathcal{I}^{\mathtt{H}}_{O,\mathcal{R}}$. We define $[\mathsf{I}, \mathsf{J}]_{O,\mathcal{R}} = \{\mathsf{I}' \in \mathcal{I}^{\mathtt{H}}_{O,\mathcal{R}}, \ \mathsf{I} \leq \mathsf{I}' \leq \mathsf{J}\}$.

Let V be a vocabulary and let r be an r-ERDF rule. We denote by $[r]_V$ the set of rules that result from r if we replace each variable $x \in FVar(r)$ by $v(x)$, for all mappings $v : FVar(r) \rightarrow V$. Let P be an r-ERDF program. We define $[P]_V = \bigcup_{r \in P} [r]_V$.

In [2], we defined the $\#n$-stable models of an ERDF ontology (for an $n \in I\!N$), based on the coherent stable models of partial logic [7] (which, on ELPs, are equivalent [7] to Answer Sets [5]). Here, we extend this definition to modular stable models of an r-ERDF ontology w.r.t. a modular ERDF ontology.

Definition 13 (Modular ERDF stable model). Let $O \in \mathcal{R}$, and let $\mathsf{M} \in \mathcal{I}^{\mathtt{H}}_{O,\mathcal{R}}$. We say that M is a *modular* (*ERDF*) *stable model* of O w.r.t. $\mathcal{R}$ iff there is a chain of modular Herbrand interpretations of O w.r.t. $\mathcal{R}$, $\mathsf{I}_0 \leq ... \leq \mathsf{I}_k$ such that $\mathsf{I}_{k-1} = \mathsf{I}_k = \mathsf{M}$ and:

1. $\mathsf{I}_0 \in minimal(\{\mathsf{I} \in \mathcal{I}^{\mathtt{H}}_{O,\mathcal{R}} \mid \ \forall O' \in D^{\mathcal{R}}_O,\ \langle \mathsf{I}, O' \rangle \models sk(G_{O'})\})$.
2. For $0 < \alpha \leq k$:
   $\mathsf{I}_\alpha \in minimal(\{\mathsf{I} \in \mathcal{I}^{\mathtt{H}}_{O,\mathcal{R}} \mid \mathsf{I} \geq \mathsf{I}_{\alpha-1}$ and $\forall O' \in D^{\mathcal{R}}_O$, it holds that:
   if $r \in [P_{O'}]_{V_{O',\mathcal{R}}}$ s.t. $\langle \mathsf{J}, O' \rangle \models Cond(r)$, $\forall \mathsf{J} \in [\mathsf{I}_{\alpha-1}, \mathsf{M}]_{O,\mathcal{R}}$, then $\langle \mathsf{I}, O' \rangle \models Concl(r)\})$.

The set of modular stable models of O w.r.t. $\mathcal{R}$ is denoted by $\mathcal{M}^{\mathtt{st}}_{O,\mathcal{R}}$. $\square$

Example 6. Consider the modular ERDF ontology $\mathcal{R}$ of Example 1. For every $\mathsf{M} \in \mathcal{M}^{\mathtt{st}}_{O_5,\mathcal{R}}$, it holds $\langle \mathsf{M}, O_1 \rangle \models \neg$ `rdf:type(geo:Egypt,geo:Europ_Country)`. This is due to the local CWA in P_{O_1}. Now, since O_5 imports class `geo:Europ_Country` from O_1, for every $\mathsf{M} \in \mathcal{M}^{\mathtt{st}}_{O_5,\mathcal{R}}$, it holds $\langle \mathsf{M}, O_5 \rangle \models \neg$ `rdf:type(geo:Egypt,geo:Europ_Country)`. Therefore, due to the 2nd rule of P_{O_5}, for every $\mathsf{M} \in \mathcal{M}^{\mathtt{st}}_{O_5,\mathcal{R}}$, $\langle \mathsf{M}, O_5 \rangle \models$ `ann:choose_trav_package(pyr:package1,geo:Egypt)`.

Note the ERDF triple `rdf:type(eu:CountryEU,erdf:TotalClass)` in G_{O_2}, expressing a local OWA. Therefore, in some $\mathsf{M} \in \mathcal{M}^{\mathtt{st}}_{O_5,\mathcal{R}}$, it holds $\langle \mathsf{M}, O_2 \rangle \models \neg$ `rdf:type(geo:Croatia,eu:CountryEU)`, while in the rest $\mathsf{M}' \in \mathcal{M}^{\mathtt{st}}_{O_5,\mathcal{R}}$, it holds $\langle \mathsf{M}', O_2 \rangle \models$ `rdf:type(geo:Croatia,eu:CountryEU)`. Now note that O_5 imports class `eu:CountryEU` from O_2. Thus, in some $\mathsf{M} \in \mathcal{M}^{\mathtt{st}}_{O_5,\mathcal{R}}$, it holds $\langle \mathsf{M}, O_5 \rangle \models \neg$ `rdf:type(geo:Croatia,eu:CountryEU)`, while in the rest $\mathsf{M}' \in \mathcal{M}^{\mathtt{st}}_{O_5,\mathcal{R}}$, it holds $\langle \mathsf{M}', O_5 \rangle \models$ `rdf:type(geo:Croatia,eu:CountryEU)`. Reasoning now by cases and due to the 3rd and 4th rule of P_{O_5}, for every $\mathsf{M} \in \mathcal{M}^{\mathtt{st}}_{O_5,\mathcal{R}}$, it holds $\langle \mathsf{M}, O_5 \rangle \models$ `ann:choose_trav_package(trav:package2,geo:Croatia)`. $\square$

The following proposition shows that any modular stable model of O w.r.t. $\mathcal{R}$ is a modular Herbrand model of O w.r.t. $\mathcal{R}$.

Proposition 1. Let $O \in \mathcal{R}$. If $\mathsf{M} \in \mathcal{M}^{\mathtt{st}}_{O,\mathcal{R}}$ then $\mathsf{M} \in \mathcal{M}^{\mathtt{H}}_{O,\mathcal{R}}$.

On the other hand, if all properties of $O' \in D^{\mathcal{R}}_O$ are total, a modular Herbrand model M of O w.r.t. $\mathcal{R}$ is a modular stable model of O w.r.t. $\mathcal{R}$.

Proposition 2. Let $O \in \mathcal{R}$. If *rdfs*:*subclass*(*rdf*:*Property*, *erdf*:*TotalProperty*) $\in G_{O'}$, for all $O' \in D^{\mathcal{R}}_O$, then $\mathcal{M}^{\mathtt{st}}_{O,\mathcal{R}} = \mathcal{M}^{\mathtt{H}}_{O,\mathcal{R}}$.

The following proposition relates the modular stable models of different `r`-ERDF ontologies w.r.t. a modular ERDF ontology.

Proposition 3. Let $O \in \mathcal{R}$ and let $O' \in D^{\mathcal{R}}_O$. Let $\mathsf{M} \in \mathcal{I}^{\mathtt{H}}_{O,\mathcal{R}}$ and let $\mathsf{M}' = \{M_{O''} \in \mathsf{M} \mid O'' \in D^{\mathcal{R}}_{O'}\}$. It holds that: If $\mathsf{M} \in \mathcal{M}^{\mathtt{st}}_{O,\mathcal{R}}$ then $\mathsf{M}' \in \mathcal{M}^{\mathtt{st}}_{O',\mathcal{R}}$. $\square$

Let $O \in \mathcal{R}$. We say that O is *inconsistent* under the modular stable model semantics w.r.t. $\mathcal{R}$ iff $\mathcal{M}^{\mathtt{st}}_{O,\mathcal{R}} = \{\}$.

Let $O \in \mathcal{R}$, and let $O' \in D^{\mathcal{R}}_O$. It follows directly from Proposition 3 that if O' is inconsistent under the modular stable model semantics w.r.t. $\mathcal{R}$ then O is also inconsistent under the modular stable model semantics w.r.t. $\mathcal{R}$. Obviously, due to the definition of $D^{\mathcal{R}}_O$ in Definition 7, all other `r`-ERDF ontologies in $\mathcal{R}$ remain unaffected from the local inconsistency.

Definition 14 (Modular ERDF stable model entailment). Let $O \in \mathcal{R}$. Additionally, let F be an $\mathtt{r}$-ERDF formula. We say that O *entails* F w.r.t. $\mathcal{R}$ under the *modular ERDF stable model semantics*, denoted by $O \models^{\mathtt{st}}_{\mathcal{R}} F$ iff for all $\mathsf{M} \in \mathcal{M}^{\mathtt{st}}_{O,\mathcal{R}}$, $\langle \mathsf{M}, O \rangle \models F$. □

The following proposition shows that modular stable model entailment extends $\#n$-stable model entailment from ERDF ontologies to modular ERDF ontologies.

Proposition 4. Let $O = \langle G, P \rangle$ be an ERDF ontology and let F be an ERDF formula. Additionally, let O' be an $\mathtt{r}$-ERDF ontology such that $G_{O'} = G$, $P_{O'} = P$, and $Int_{O'} = \{\}$. It holds: $O \models^{\mathtt{st}\#n_{\mathcal{R}}} F$ iff $O' \models^{\mathtt{st}}_{\mathcal{R}} F$, where $\mathcal{R} = \{O'\}$.

The following corollary follows directly from the above proposition and Proposition 3 in [2], and it shows that modular stable model entailment extends RDFS entailment from RDF graphs to modular ERDF ontologies.

Corollary 1. Let G, G' be RDF graphs such that $V_G \cap \mathcal{V}_{ERDF} = \emptyset$, $V_{G'} \cap \mathcal{V}_{ERDF} = \emptyset$, and $V_{G'} \cap sk_G(Var(G)) = \emptyset$. Let O be an $\mathtt{r}$-ERDF ontology with $G_O = G$, $P_O = \{\}$, and $Int_O = \{\}$. If $max(\{i \in I\!N \mid rdf{:}\_i \in V_{G'}\}) \leq n_{\mathcal{R}}$ then: $G \models^{RDFS} G'$ iff $O \models^{\mathtt{st}}_{\mathcal{R}} G'$, where $\mathcal{R} = \{O\}$.

Let $O \in \mathcal{R}$ and let F be an $\mathtt{r}$-ERDF formula. The *modular stable answers* of F w.r.t. O and $\mathcal{R}$ are defined as follows[9]:

$$Ans^{\mathtt{st}}_{O,\mathcal{R}}(F) = \begin{cases} \text{“yes”} & \text{if } FVar(F) = \emptyset \text{ and } O \models^{\mathtt{st}}_{\mathcal{R}} F \\ \text{“no”} & \text{if } FVar(F) = \emptyset \text{ and } O \not\models^{\mathtt{st}}_{\mathcal{R}} F \\ \{v : FVar(F) \rightarrow V_{O,\mathcal{R}} \mid O \models^{\mathtt{st}}_{\mathcal{R}} v(F)\}, & \text{if } FVar(F) \neq \emptyset \end{cases}$$

Example 7. Consider the modular ERDF ontology $\mathcal{R}$ of Example 1. Then: $Ans^{\mathtt{st}}_{O_5,\mathcal{R}}(\mathtt{ann{:}choose\_trav\_package}(?x, ?y)) = \{\langle ?x = \mathtt{pyr{:}package1}, ?y = \mathtt{geo{:}Egypt}\rangle, \langle ?x = \mathtt{trav{:}package2}, ?y = \mathtt{geo{:}Croatia}\rangle\}$. □

An $\mathtt{r}$-ERDF formula is called *simple*, if it has the form: $t_1 \wedge ... \wedge t_k \wedge {\sim} t_{k+1} \wedge ... \wedge {\sim} t_n \wedge ({\sim} t'_1)@Nam_{O_1} \wedge ... \wedge ({\sim} t'_m)@Nam_{O_m}$, where (i) each t_i is a normal or qualified ERDF triple (positive or negative), (ii) each t'_i is a normal ERDF triple (positive or negative), and (iii) each O_i is an $\mathtt{r}$-ERDF ontology. An $\mathtt{r}$-ERDF program P is called *simple* if the body of each rule in P is simple, or $\mathtt{true}$. Let $\mathcal{R}$ be a modular ERDF ontology and let $O \in \mathcal{R}$. O is called *simple* w.r.t. $\mathcal{R}$, if for each $O' \in D^{\mathcal{R}}_O$, $P_{O'}$ is simple.

Let $O \in \mathcal{R}$ s.t. O is simple w.r.t. $\mathcal{R}$. We can show that the modular stable answers of a simple $\mathtt{r}$-ERDF formula F w.r.t. O and $\mathcal{R}$ can be computed through Answer Set Programming [5] on an ELP $\Pi_{O,\mathcal{R}}$.

Below, we state several complexity results of the modular ERDF stable model semantics. We define $size\_inst(O, \mathcal{R}) = \sum\{\text{size of } (G_{O'} \cup [P_{O'}]_{V_{O',\mathcal{R}}}) \mid O' \in D^{\mathcal{R}}_O\}$.

Proposition 5. Let $O \in \mathcal{R}$, and let F be an $\mathtt{r}$-ERDF formula. Additionally, let v be (i) one of {“yes”, “no”}, if $Var(F) = \emptyset$, or (ii) a mapping $v : Var(F) \rightarrow V_{O,\mathcal{R}}$, if $Var(F) \neq \emptyset$.

[9] $v(F)$ is the formula F after replacing all the free variables x in F by $v(x)$.

1. If O is a simple $\mathtt{r}$-ERDF ontology w.r.t. $\mathcal{R}$ then: (i) the problem of establishing whether O has a modular stable model w.r.t. $\mathcal{R}$ is *NP*-complete w.r.t. $size\_inst(O, \mathcal{R})$, and (ii) the problem of establishing whether $v \in Ans^{\mathtt{st}}_{O,\mathcal{R}}(F)$ is *co-NP*-complete w.r.t. $size\_inst(O, \mathcal{R})$.
2. If for all $O' \in D^{\mathcal{R}}_{O}$, no quantifies $\forall$, $\exists$ appear in $P_{O'}$ then: (i) the problem of establishing whether O has a modular stable model w.r.t. $\mathcal{R}$ is $\Sigma^P_2 = NP^{NP}$-complete w.r.t. $size\_inst(O, \mathcal{R})$, and (ii) the problem of establishing whether $v \in Ans^{\mathtt{st}}_{O,\mathcal{R}}(F)$ is $\Pi^P_2 = co\text{-}NP^{NP}$-complete w.r.t. $size\_inst(O, \mathcal{R})$.
3. In the *general* case, (i) the problem of establishing whether O has a modular stable model w.r.t. $\mathcal{R}$ is *PSPACE*-complete w.r.t. $size\_inst(O, \mathcal{R})$, and (ii) the problem of establishing whether $v \in Ans^{\mathtt{st}}_{O,\mathcal{R}}(F)$ is *PSPACE*-complete w.r.t. $size\_inst(O, \mathcal{R})$.

5 Conclusions & Related Work

In this paper, we extended ERDF ontologies [1], and thus RDF graphs to `r`-ERDF ontologies. In particular, an `r`-ERDF ontology is an ERDF ontology that (i) is associated with a set of export and import statements, and (ii) interacts with other `r`-ERDF ontologies (through qualified ERDF formulas in the program rules). Further, we defined a modular ERDF ontology as a set of `r`-ERDF ontologies and defined its modular stable model semantics, model-theoretically, based on partial logic [7]. We showed that modular stable model entailment on modular ERDF ontologies extends $\#n$-stable model entailment on ERDF ontologies [2], and thus it also extends RDFS entailment on RDF graphs [6]. Future work concerns (i) the extension of the modular stable model semantics such that meaning is assigned to inconsistent `r`-ERDF ontologies of a modular ERDF ontology, and (ii) the implementation of the modular ERDF framework.

N3Logic [3] allows rules to be integrated with RDF. Indeed, part of the RDFS semantics is represented by program rules. Yet, the supported form of negation as failure, expressed through the built-in `log:notincludes`, is limited. Additionally, N3Logic does not have a model-theoretic semantics that faithfully extends RDFS semantics [6], does not support explicit negation and general formulas in the body of the rules, and ignores visibility issues.

A modularity framework for RDF rule bases (without blank nodes) is proposed in [11]. There, RDFS semantics are partially represented through a normal logic program, associated with a special context/module c_{RDFS}. The *contextually closed* `AS` and *contextually closed* `WFS` semantics of such a modular RDF rule base $\mathcal{R}$ are defined, through the `AS` [5] and `WFS` [4] semantics of a normal logic program $\mathcal{R}_{CC}$, generated from $\mathcal{R}$, respectively. Yet, this framework does not have a model-theoretic semantics that faithfully extends RDFS semantics [6], does not support explicit negation and general formulas in the body of the rules, and ignores visibility issues.

TRIPLE [12] is a rule language for the Semantic Web that supports modules (called, *models* there), qualified literals, and dynamic module transformation. Arbitrary formulas can be used in the body of a rule, handled through the Lloyd-Topor transformations [9]. Part of the semantics of the RDF(S) vocabulary is represented as pre-defined rules (and not as semantic conditions on interpretations), which are grouped together in a module. The semantics of a modular rule base is defined, based on the *well-founded semantics* (`WFS`) [4] of an equivalent logic program. Yet, the model-theoretic semantics of TRIPLE [13] does not faithfully extend RDFS semantics [6] and is not, in general, equivalent to its transformational semantics. Additionally, TRIPLE does not support explicit negation and ignores visibility issues.

References

1. A. Analyti, G. Antoniou, C. V. Damásio, and G. Wagner. Extended RDF as a Semantic Foundation of Rule Markup Languages. *Journal of Artificial Intelligence Research (JAIR)*, 32:37–94, 2008.
2. A. Analyti, G. Antoniou, C. V. Damásio, and G. Wagner. On the Computability and Complexity Issues of Extended RDF. In *10th Pacific Rim International Conference on Artificial Intelligence (PRICAI-2008)*, pages 5–16, 2008.
3. T. Berners-Lee, D. Connolly, L. Kagal, Y. Scharf, and J. Hendler. N3Logic: A Logical Framework For the World Wide Web. *TPLP*, 8(3):249–269, 2008.
4. A. V. Gelder, K. A. Ross, and J. S. Schlipf. The Well-Founded Semantics for General Logic Programs. *Journal of the ACM*, 38(3):620–650, 1991.
5. M. Gelfond and V. Lifschitz. Logic programs with Classical Negation. In *7th International Conference on Logic Programming*, pages 579–597, 1990.
6. P. Hayes. RDF Semantics. W3C Recommendation, 10 February 2004. Available at `http://www.w3.org/TR/2004/REC-rdf-mt-20040210/`.
7. H. Herre, J. Jaspars, and G. Wagner. Partial Logics with Two Kinds of Negation as a Foundation of Knowledge-Based Reasoning. In D. M. Gabbay and H. Wansing, editors, *What Is Negation?* Kluwer Academic Publishers, 1999.
8. G. Klyne and J. J. Carroll. Resource Description Framework (RDF): Concepts and Abstract Syntax. W3C Recommendation, 10 February 2004. Available at `http://www.w3.org/TR/2004/REC-rdf-concepts-20040210/`.
9. J. W. Lloyd and R. W. Topor. Making Prolog more Expressive. *Journal of Logic Programming*, 1(3):225–240, 1984.
10. E. Oikarinen and T. Janhunen. Achieving compositionality of the stable model semantics for smodels programs. *TPLP*, 8(5–6):717–761, 2008.
11. A. Polleres, C. Feier, and A. Harth. Rules with Contextually Scoped Negation. In *3rd European Semantic Web Conference (ESWC-2006)*, pages 332–347, 2006.
12. M. Sintek and S. Decker. TRIPLE - A Query, Inference, and Transformation Language for the Semantic Web. In *1st International Semantic Web Conference (ISWC-2002)*, pages 364–378. Springer-Verlag, 2002.
13. W. N. Stefan Decker, Michael Sintek. The Model-Theoretic Semantics of TRIPLE. Technical Report, 2002.
14. G. Wagner, A. Giurca, I.-M. Diaconescu, G. Antoniou, A. Analyti, and C. Damasio. Reasoning on the Web with Open and Closed Predicates. In *3rd International Workshop on Applications of Logic Programming to the (Semantic) Web and Web Services (ALPSWS 2008), in conjunction with ICLP'08*, pages 71–84, 2008.