# The Resource Description Framework RDF

Jos de Bruijn

October 17, 2007

The RDF semantics specification [Hay04] defines simple, RDF, and RDFS entailment. The differences between the kinds of entailment lie in the way the RDF and RDFS vocabularies are interpreted; in simple entailment the RDF(S) vocabularies are not interpreted and in RDF entailment the RDFS vocabulary is not interpreted.

We now proceed with a review of the formal definitions the RDF syntax and semantics.

A vocabulary $V = \langle \mathcal{F}, \mathcal{PL}, \mathcal{TL} \rangle$ consists of a set $\mathcal{F}$ of RDF URI references (which are effectively absolute IRIs [DS05]; we will refer to them simply as *URIs* in the remainder), a set $\mathcal{PL}$ of plain literals (i.e. Unicode character strings with an optional language tag), and a set $\mathcal{TL}$ of typed literals (i.e. pairs $(s, u)$ of a Unicode string $s$ and a URI $u$, denoting a datatype); see [KC04, Sections 6.4, 6.5, 6.6] for more details about the specific form of these symbols. Note that $\mathcal{F}$, $\mathcal{PL}$, and $\mathcal{TL}$ are mutually disjoint.

Let $\mathcal{B}$ be a set of blank nodes (i.e. existentially quantified variables) which is disjoint from all sets of identifiers in $V$. Terms are RDF URI references, plain or typed literals, or blank nodes. An *RDF graph* $S$ is a set of triples $\langle s, p, o \rangle$, with $s \in \mathcal{F} \cup \mathcal{B}$, $p \in \mathcal{F}$, and $o \in \mathcal{F} \cup \mathcal{PL} \cup \mathcal{TL} \cup \mathcal{B}$. A *ground triple* is a triple which does not contain blank nodes. A *ground* RDF graph is a set of ground triples. With $bl(\langle s, p, o \rangle) \subseteq \mathcal{B}$ (resp., $bl(S) \subseteq \mathcal{B}$) we denote the set of blank nodes in a triple $\langle s, p, o \rangle$ (resp., graph $S$).

An interpretation is a tuple $I = \langle IR, IP, LV, IS, IL, IEXT \rangle$, where $IR$ is a countable, non-empty set, called the domain, $IP$ is a countable set of properties, $LV \subseteq IR$ is a countable set of literal values with $\mathcal{PL} \subseteq LV$, $IS$ is a mapping $IS : \mathcal{F} \to IR \cup IP$, $IL$ is a mapping $IL : \mathcal{TL} \to IR$, and $IEXT$ is an extension function $IEXT : IP \to 2^{(IR \times IR)}$.

Given an interpretation $I$ and a subset of the blank nodes $\mathcal{B}' \subseteq \mathcal{B}$, we define a mapping $A : \mathcal{B}' \to IR$ which is used to interpret blank nodes. For a term $t$ we define $t^{I,A}$ as:

- if $t \in \mathcal{F}$, then $t^{I,A} = IS(t)$,
- if $t \in \mathcal{PL}$, then $t^{I,A} = t$,
- if $t \in \mathcal{TL}$, then $t^{I,A} = IL(t)$, and
- if $t \in \mathcal{B}'$, then $t^{I,A} = A(t)$.

An interpretation $I$ *satisfies* a triple $\langle s, p, o \rangle$ with respect to a mapping $A : \mathcal{B}' \to IR$, with $bl(\langle s, p, o \rangle) \subseteq \mathcal{B}'$, denoted $(I, A) \models \langle s, p, o \rangle$, if $p^{I,A} \in IP$

and $\langle s^{I,A}, o^{I,A}\rangle \in IEXT(p^{I,A})$. $I$ satisfies a graph $S$ with respect to a mapping $A : bl(S) \to IR$, denoted $(I, A) \models S$, if $(I, A) \models \langle s, p, o\rangle$ for every $\langle s, p, o\rangle \in S$.

An interpretation $I$ satisfies (i.e. is a *model* of) an RDF graph $S$, denoted $I \models S$, if $(I, A) \models S$ for some $A : bl(S) \to IR$.

Any interpretation is an *s-interpretation* (simple interpretation). An RDF graph $S$ *s-entails* an RDF graph $E$, denoted $S \models_s E$, if every model of $S$ is a model of $E$.

**RDF Interpretations** The RDF vocabulary consists of the following symbols[1]:

> `type` `Property` `XMLLiteral` `nil` `List` `Statement` `subject` `predicate`
> `object` `first` `rest` `Seq` `Bag` `Alt` `_1` `_2` ... `value`

Note that `_i`, for any positive integer `i` > 0, is part of the RDF vocabulary. Thus, the RDF vocabulary is infinite.

The *RDF ontology vocabulary* is that part of the RDF vocabulary which is used for defining ontological notions, and consists of the classes `XMLLiteral` and `Property` of XML literals and properties, respectively, and the property `type`, corresponds to the class membership relation.

Given an interpretation $I$, the *class extension* of an object $x \in IR$ is the set of elements connected to $x$ via `type` (i.e. the instances of $x$): $ICEXT(x) = \{k \mid \langle k, x\rangle \in IEXT(IS(\texttt{type}))\}$.
An interpretation $I$ of a vocabulary $V = \langle \mathcal{F}, \mathcal{PL}, \mathcal{TL}\rangle$ is an *rdf-interpretation* if $V$ includes the RDF vocabulary and the conditions depicted in Table 1 hold. In the table, $xml(s)$ denotes the XML value corresponding to $s$, as defined in [KC04].

| | |
|---|---|
| 1 | $IS(\texttt{type}), IS(\texttt{subject}), IS(\texttt{predicate}), IS(\texttt{object}), IS(\texttt{first}), IS(\texttt{rest}),$ $IS(\texttt{value}), IS(\texttt{\_1}), IS(\texttt{\_2}), \ldots \in IP$ |
| | $IS(\texttt{nil}) \in ICEXT(IS(\texttt{List}))$ |
| 2 | $IP = ICEXT(IS(\texttt{Property}))$ |
| 3 | if $(s, \texttt{XMLLiteral}) \in \mathcal{TL}$ and $s$ is valid XML, then $IL((s, \texttt{XMLLiteral})) = xml(s)$, $IL((s, \texttt{XMLLiteral})) \in LV$, and $IL((s, \texttt{XMLLiteral})) \in ICEXT(IS(\texttt{XMLLiteral}))$ |
| 4 | if $(s, \texttt{XMLLiteral}) \in \mathcal{TL}$ and $s$ is not valid XML, then $IL((s, \texttt{XMLLiteral})) \notin LV$ and $IL((s, \texttt{XMLLiteral})) \notin ICEXT(IS(\texttt{XMLLiteral}))$ |

Table 1: Conditions on RDF interpretations

Notice that condition 1 corresponds to the RDF axiomatic triples in [Hay04, Section 3.1] in the following way: a condition $IS(s) \in IP$ corresponds to an axiomatic triple $\langle s$, `type`, `Property`$\rangle$; a condition $IS(s) \in ICEXT(IS(o))$ corresponds to an axiomatic triple $\langle s$, `type`, $o\rangle$.

[1]Note that, for brevity, we leave out the usual RDF namespace prefix.

**RDFS Interpretations** The RDFS vocabulary consists of (again, we omit the usual RDFS namespace prefix):

```
domain range Resource Literal Datatype Class subClassOf
subPropertyOf member Container ContainerMembershipProperty
comment seeAlso isDefinedBy label
```

The *RDFS ontology vocabulary*, analogous to the RDF ontology vocabulary, consists of the classes `Resource`, `Literal`, `Datatype`, `Class` and `ContainerMembershipProperty` of resources, literals, datatypes, classes and membership properties, respectively, and properties `domain`, `range`, `subClassOf`, and `subPropertyOf` for property domain and range restrictions, and subclass and subproperty relations, respectively. When referring to the RDF(S) ontology vocabulary we mean the union of the RDF and RDFS ontology vocabularies.

We say that an rdf-interpretation $I$ of vocabulary $V$ is an *rdfs-interpretation* if $V$ includes the RDFS vocabulary, and the conditions on the RDFS interpretations depicted in Table 2 hold. As a shortcut, we define $I_{ext(p)}(o) = \{s \mid \langle s, IS(o)\rangle \in I_{ext}(IS(p))\}$.

Note that condition 5 in Table 2 corresponds to the RDFS axiomatic triples in [Hay04, Section 4.1] in the following way: a condition $IS(s) \in I_{ext(p)}(o)$ corresponds to an axiomatic triple $\langle s, p, o\rangle$, and a condition $IS(s) \in ICEXT(IS(c))$ corresponds to an axiomatic triple $\langle s, \texttt{type}, c\rangle$.

**Entailment and Satisfiability** Given a vocabulary $V$ and an entailment regime $x \in \{s, rdf, rdfs\}$, a generalized (resp., standard) RDF graph $S$ *x-entails* a generalized (resp., standard) RDF graph $E$, denoted $S \models_x E$, if every $x$-interpretation of $V$ which is a model of $S$ is a model of $E$.

Given an entailment regime $x \in \{s, rdf, rdfs\}$, an RDF graph $S$ is *x-satisfiable* if it has a model; otherwise it is *x-unsatisfiable*. The following observations can be made about satisfiability for the various entailment regimes.

**Proposition 1** ([Hay04])**.**

1. *Every RDF graph is s-satisfiable.*
2. *Every RDF graph is rdf-satisfiable.*
3. *There is an RDF graph which is not rdfs-satisfiable.*

# References


[DS05] M. Duerst and M. Suignard. Internationalized resource identifiers (iris). Proposed standard RFC 3987, Internet Engineering Task Force, 2005.

[Hay04] Patrick Hayes. RDF semantics. Recommendation 10 February 2004, W3C, 2004.

[KC04] Graham Klyne and Jeremy J. Carroll. Resource description framework (RDF): Concepts and abstract syntax. Recommendation 10 February 2004, W3C, 2004.

| | |
|---|---|
| 5 | $IS(\texttt{type}), IS(\texttt{member}), IS(\texttt{seeAlso}), IS(\texttt{isDefinedBy}), IS(\texttt{comment}),$ $IS(\texttt{label}), IS(\texttt{value}), IS(\_1), IS(\_2), \ldots \in I_{ext(\texttt{domain})}(\texttt{Resource})$ |
| | $IS(\texttt{domain}), IS(\texttt{range}), IS(\texttt{subPropertyOf}) \in I_{ext(\texttt{domain})}(\texttt{Property})$ |
| | $IS(\texttt{subClassOf}) \in I_{ext(\texttt{domain})}(\texttt{Class})$ |
| | $IS(\texttt{subject}), IS(\texttt{predicate}), IS(\texttt{object}) \in I_{ext(\texttt{domain})}(\texttt{Statement})$ |
| | $IS(\texttt{first}), IS(\texttt{rest}) \in I_{ext(\texttt{domain})}(\texttt{List})$ |
| | $IS(\texttt{subject}), IS(\texttt{predicate}), IS(\texttt{object}), IS(\texttt{member}), IS(\texttt{first}), IS(\texttt{seeAlso}),$ $IS(\texttt{isDefinedBy}), IS(\texttt{value}), IS(\_1), IS(\_2), \ldots \in I_{ext(\texttt{range})}(\texttt{Resource})$ |
| | $IS(\texttt{comment}), IS(\texttt{label}) \in I_{ext(\texttt{range})}(\texttt{Literal})$ |
| | $IS(\texttt{subPropertyOf}) \in I_{ext(\texttt{range})}(\texttt{Property})$ |
| | $IS(\texttt{type}), IS(\texttt{domain}), IS(\texttt{range}), IS(\texttt{subClassOf}) \in I_{ext(\texttt{range})}(\texttt{Class})$ |
| | $IS(\texttt{rest}) \in I_{ext(\texttt{range})}(\texttt{List})$ |
| | $IS(\texttt{Alt}), IS(\texttt{Bag}), IS(\texttt{Seq}) \in I_{ext(\texttt{subClassOf})}(\texttt{Container})$ |
| | $IS(\texttt{ContainerMembershipProperty}) \in I_{ext(\texttt{subClassOf})}(\texttt{Property})$ |
| | $IS(\texttt{isDefinedBy}) \in I_{ext(\texttt{subPropertyOf})}(\texttt{seeAlso})$ |
| | $IS(\texttt{XMLLiteral}) \in ICEXT(IS(\texttt{Datatype}))$ |
| | $IS(\texttt{XMLLiteral}) \in I_{ext(\texttt{subClassOf})}(\texttt{Literal})$ |
| | $IS(\texttt{Datatype}) \in I_{ext(\texttt{subClassOf})}(\texttt{Class})$ |
| | $IS(\_1), IS(\_2), \ldots \in ICEXT(IS(\texttt{ContainerMembershipProperty}))$ |
| 6 | $IR = ICEXT(IS(\texttt{Resource}))$ $LV = ICEXT(IS(\texttt{Literal}))$ |
| 7 | if $\langle x, y\rangle \in IEXT(IS(\texttt{domain}))$ and $\langle u, v\rangle \in IEXT(x)$, then $u \in ICEXT(y)$ |
| 8 | if $\langle x, y\rangle \in IEXT(IS(\texttt{range}))$ and $\langle u, v\rangle \in IEXT(x)$, then $v \in ICEXT(y)$ |
| 9 | $IEXT(IS(\texttt{subPropertyOf}))$ is transitive and reflexive on $IP$ |
| 10 | if $\langle x, y\rangle \in IEXT(IS(\texttt{subPropertyOf}))$, then $IEXT(x) \subseteq IEXT(y)$ |
| 11 | if $x \in ICEXT(\texttt{Class})$, then $\langle x \in I_{ext(\texttt{subClassOf})}(\texttt{Resource})$ |
| 12 | if $\langle x, y\rangle \in IEXT(IS(\texttt{subClassOf}))$, then $ICEXT(x) \subseteq ICEXT(y)$ |
| 13 | $IEXT(IS(\texttt{subClassOf}))$ is transitive and reflexive on $ICEXT(\texttt{Class})$ |
| 14 | if $x \in ICEXT(\texttt{ContainerMembershipProperty})$, then $x \in I_{ext(\texttt{subPropertyOf})}(\texttt{member})$ |
| 15 | if $x \in ICEXT(\texttt{Datatype})$, then $x \in I_{ext(\texttt{subClassOf})}(\texttt{Literal})$ |

Table 2: Conditions on RDFS interpretations