Integrity Constraints for Linked Data

Alan Jeffrey and Peter F. Patel–Schneider

Alcatel–Lucent Bell Labs

Abstract. Linked Data makes one central addition to the Semantic Web principles: all entity URIs should be dereferenceable to provide an authoritative RDF representation. URIs in a linked dataset can be partitioned into the exported URIs for which the dataset is authoritative versus the imported URIs the dataset is linking against. This partitioning has an impact on integrity constraints, as a Closed World Assumption applies to the exported URIs, while a Open World Assumption applies to the imported URIs. We provide a definition of integrity constraint satisfaction in the presence of partitioning, and show that it leads to a formal interpretation of dependency graphs which describe the hyperlinking relations between datasets. We prove that datasets with integrity constraints form a symmetric monoidal category, from which the soundness of acyclic dependency graphs follows.

1 Introduction

Motivation. In the Semantic Web, entities are named by URIs, and are described by RDF documents. Linked Data [5] adds the constraint that entity URIs should be *dereferenceable* (HTTP URIs which accept `GET` requests), and dereferencing an entity URI returns an RDF *representation* of that entity. The W3C web architecture [8] calls such representations *authoritative*.

The RDF triples contained in an entity representation will generally refer to entities for which the representation is not authoritative. Such hyperlinks between datasets are often visualized as a *dependency graph*, such as the popular Linking Open Data cloud diagram [6] shown in Figure 1.

Linked Data puts a new spin on the open world stance of the Semantic Web: from the point of view of a given URI owner, the world is partitioned into local entities, for which the owner is authoritative, and imported entities, for which the owner is not authoritative. In this paper, we provide a formal model of this partitioning which includes:

- a partitioning of entities into *imported* and *exported* nodes, in addition to the familiar *blank* nodes,
- a definition of what it means for a dataset to satisfy its integrity constraints, based on the minimal models of Motik *et al.* [13], but adapted to partitioning,
- a model of acyclic dependency graphs, which can be built compositionally, and where integrity constraint satisfaction can be performed locally, and
- a proof that graphical reasoning for datasets is sound, by showing that datasets form a symmetric monoidal category.

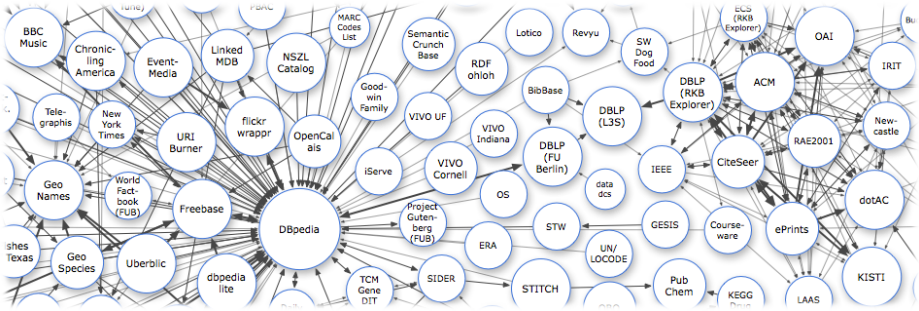


Fig. 1. Linking Open Data cloud diagram (detail)

This paper gives the first formal treatment of authoritative resource, integrity constraints for linked data, dependency graphs, and the categorical structure of semantic data. In this introduction, we provide informal examples to motivate our model, which are made precise in Sections 4 and 5.

Authoritative representations, imports and exports. The W3C web architecture [8] recommends as good practice that a URI owner "should provide authoritative representations of the resource it identifies". Typically, these are HTTP URIs, which respond to `GET` requests. Linked Data [5] applies this practice to the Semantic Web: URI owners provide authoritative representations of their URIs in RDF (for datasets) or OWL (for ontologies).

Semantic reasoners can make deductions from Linked Data. For example, consider a URI `bob:` (in examples, we will use URI prefixes such as `alice:` and `bob:`) which dereferences to the Turtle [4] representation:

```
bob: foaf:primaryTopic bob:me .
bob:me foaf:knows [ foaf:homepage alice: ] .
```

Now, if `alice:` dereferences to:

```
alice: foaf:primaryTopic alice:me .
```

then a reasoner can deduce (using the FOAF [2] specification's definitions):

```
bob:me foaf:knows alice:me .
```

In Linked Data, the entities in a dataset can be partitioned into:

- *exported nodes* (*enodes*): local entities, which the representation is authoritative for, with a publicly defined name that other datasets may link against,
- *blank nodes* (*bnodes*): local entities, which the representation is authoritative for, but without a publicly defined name, and
- *imported nodes* (*inodes*): all other entities.

For example, the RDF representation of `bob:` given above contains inode `alice:`, enodes `bob:` and `bob:me`, and an anonymous bnode (called `_:anon` below).

Ontologies and integrity constraints. A consumer of Linked Data may wish to assume a notion of correctness of the data it is consuming. Rather than considering integrity constraints to be given in a separate formalism such as a rules engine [14] or epistemic logic [15], we will use ontologies to express both deductive reasoning (the *standard* ontology), and the correctness criteria (the *constraint* ontology). A similar approach was taken by Motik *et al.* [13] and Tao *et al.* [17]. For example, consider the standard ontology:

$$\mathsf{homepage} \doteq \mathsf{primaryTopic}^{-}$$
$$\mathsf{PersonalHomePage} \sqsubseteq \mathsf{Document}$$
$$\mathsf{Document} \sqsubseteq \leq 1\, \mathsf{primaryTopic}$$

and the constraint ontology:

$$\mathsf{PersonalHomePage} \sqsubseteq \exists \mathsf{primaryTopic} \,.\, \mathsf{Person}$$
$$\mathsf{Person} \sqsubseteq \forall \mathsf{knows} \,.\, \mathsf{Person}$$

The above example is correct with respect to the *exported interface*:

$$\mathsf{Person}(\texttt{bob:me}) \quad \mathsf{PersonalHomePage}(\texttt{bob:})$$

under the assumption of the *imported interface*:

$$\mathsf{PersonalHomePage}(\texttt{alice:})$$

We reason informally as follows (this will be made formal in later sections).

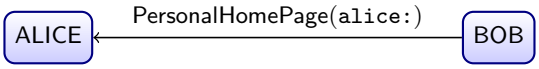
- The constraint $\mathsf{PersonalHomePage} \sqsubseteq \exists \mathsf{primaryTopic} \,.\, \mathsf{Person}$ is satisfied because the only new PersonalHomePage entity is `bob:`, and we have a witness `bob:me` for the role primaryTopic.
- The constraint $\mathsf{Person} \sqsubseteq \forall \mathsf{knows} \,.\, \mathsf{Person}$ is satisfied because the only new Person entity is `bob:me`, and the only entity which `bob:me` knows is `_:anon`. Now, in any world where $\mathsf{PersonalHomePage}(\texttt{alice:})$, there must be some individual i such that $\mathsf{primaryTopic}(\texttt{alice:}, i)$ and $\mathsf{Person}(i)$. We can then reason using the standard ontology that $i = \texttt{\_:anon}$ and so $\mathsf{Person}(\texttt{\_:anon})$.

This example, shows the use of two different styles of reasoning.

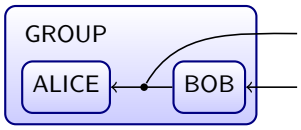
- When reasoning about exported or blank nodes, we can assume that the only properties are ones which can be deduced from information we have asserted, using the standard ontology. For example, this form of reasoning is used in "because the only new PersonalHomePage entity is `bob:`" and "the only entity in a knows role with `bob:me` is `_:anon`."
- We reason differently about imported nodes. All we know about the imported world is that it satisfies the imported interface, the standard ontology and the constraint ontology. For example, this form of reasoning is used in "in any world where $\mathsf{PersonalHomePage}(\texttt{alice:})$, there must be some individual i such that $\mathsf{primaryTopic}(\texttt{alice:}, i)$ and $\mathsf{Person}(i)$."

More succinctly, we use a Closed World Assumption for blank and exported nodes, and an Open World Assumption for imported nodes.

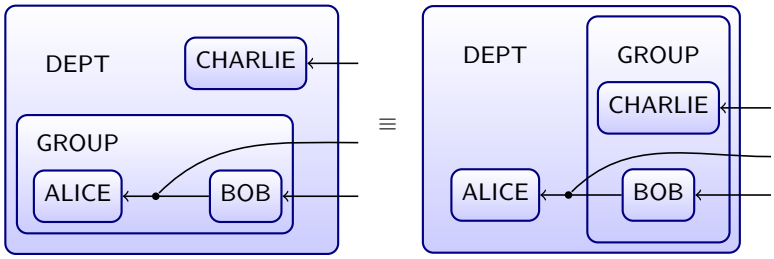
Dependency graphs. Dependency graphs such as Figure 1 are a common way of visualizing linked data, but have, until now, remained informal. We propose a formalization of such graphs as directed graphs where nodes are datasets such as ALICE (the authoritative representation of `alice:`) and BOB (the authoritative representation of `bob:`), and edges indicate the existence of hyperlinks between datasets. These edges are labeled by interfaces to make the contract between datasets explicit, for example:



Dependency graphs can be regarded as datasets, given by taking the union of all their constituent datasets (with a bit of bookkeeping to rename nodes to ensure no name clashes). Since dependency graphs form datasets, they can be nested, for example a GROUP which includes ALICE and BOB might be built:



Ensuring correctness should be compositional, for example knowing that ALICE and BOB are correct should ensure correctness of GROUP. Moreover, nested graphs should respect equivalence of datasets: if ALICE is replaced by an equivalent ALICE′, then GROUP should be equivalent to GROUP′. Finally, isomorphic graphs should be equivalent, irrespective of how they are composed, for example:



Symmetric monoidal categories. Our goals for dependency graphs are:

- Nodes describe datasets, edges describe hyperlink relationships.
- Graphs can be built compositionally, with local checking of correctness.
- Graph construction respects equivalence of datasets.
- Isomorphism of dependency graphs implies equivalence of datasets.

Proving these properties directly would be difficult, but fortunately there is an existing structure which guarantees these properties: a *symmetric monoidal category*. Category theory forms a foundational framework for mathematics, but our need of it is quite pragmatic: the equational theory of a symmetric monoidal category is precisely that of direct acyclic graphs (shown by Joyal and Street [10], see, for example, Selinger [16]). Figure 2 sketches how directed acyclic graphs form a symmetric monoidal category:

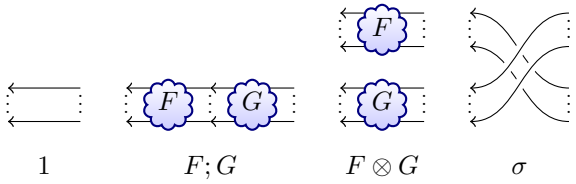


Fig. 2. Directed acyclic graphs form a (strict) symmetric monoidal category.

- The identity graph 1 just connects its source and target edges.
- The composition $F;G$ of graphs takes the disjoint union of F and G, and unifies the target edges of F with the source edges of G.
- The tensor $F \otimes G$ of graphs takes the disjoint union of F and G.
- The symmetry graph σ just permutes its source and target edges.

Since the equational theory of a symmetric monoidal category is precisely that of directed acyclic graphs, we can replace our goals for dependency graphs by the goal of showing that datasets form a symmetric monoidal category. This is a matter of proving a handful of equations, which is a easier than proving directly that graph isomorphism implies dataset equivalence.

Summary. The remainder of this paper will make this motivational section precise. We will define a notion of integrity constraint suitable for partitioning, and show that datasets with integrity constraints form a symmetric monoidal category, and hence can be formalized by dependency graphs. This is the first such investigation of integrity constraints for Linked Data. All results presented in this paper have been mechanically verified, using the Agda [1] mechanical proof assistant; all proofs are publicly available [9].

2 Preliminaries

In this paper, we consider a Description Logic $\mathcal{SHIN}_1^+$, which includes role hierarchies, role inverses, disjoint, reflexive, irreflexive and transitive roles, and singleton cardinality restrictions. We expect the results to apply to other description logics. Spelling this out, *roles* and *concepts* are defined by the grammars (where r and c are drawn from sets of atomic role names and concept names):

$$R ::= r \mid r^- \qquad C ::= c \mid \neg c \mid \bot \mid \top \mid C_1 \sqcap C_2 \mid C_1 \sqcup C_2 \mid \forall R\,.\,C \mid \exists R\,.\,C \mid \leq\!1\,R \mid >\!1\,R$$

A *TBox* is a finite set of *axioms* of the form:

$$C_1 \sqsubseteq C_2 \quad \text{or} \quad R_1 \sqsubseteq R_2 \quad \text{or} \quad \mathsf{Dis}(R_1, R_2) \quad \text{or} \quad \mathsf{Ref}(R) \quad \text{or} \quad \mathsf{Irr}(R) \quad \text{or} \quad \mathsf{Tra}(R)$$

Following Motik *et al.* [13], we assume ambient TBoxes S (the *standard* TBox) and T (the *constraint* TBox). For any finite set X, an *ABox* over X is a finite set of *assertions* of the form:

$$c(x) \quad \text{or} \quad r(x,y) \quad \text{or} \quad x \approx y \quad \text{where } x, y \in X$$

Note that ABoxes are restricted to contain only positive statements, and so have a monotone semantics. In many cases, this does not impact expressivity, as S can give names for arbitrary concepts, and T can introduce an irreflexive role differentFrom used in place of $\not\approx$ assertions. In practice, RDF is limited to positive atomic statements.

An *interpretation* $\mathcal{I}$ over X consists of a set $\Delta^{\mathcal{I}}$, together with $c^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$ for each concept name c, $r^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$ for each role name r, and $x^{\mathcal{I}} \in \Delta^{\mathcal{I}}$ for each $x \in X$. The *satisfaction* relations $\mathcal{I} \vDash A$ (for an ABox A over X) and $\mathcal{I} \vDash T$ (for a TBox T) are standard. Note that if $\mathcal{I}$ is an interpretation over $X \supseteq Y$, then $\mathcal{I}$ can be regarded as an interpretation over Y.

In the following, we will write $X \uplus Y$ for the disjoint union of X and Y: for simplicity, we will assume that X and Y are disjoint, and so $X \subseteq X \uplus Y \supseteq Y$. In the mechanized proofs [9], we use explicit tagging to ensure disjointness.

3 Initial interpretations

Consider ABoxes A over X, B over Y, and F over $(X \uplus V \uplus Y)$. We can think of A as the imported interface (where X is the set of inodes), B as the exported interface (where Y is the set of enodes) and F as the dataset (where V is the set of bnodes). Now, what does it mean for F to import A and export B, in the presence of ambient TBoxes S and T?

F can be thought of as a recipe for adding new assertions to an existing interpretation. Given any interpretation $\mathcal{I}$ over X which satisfies (S, T, A), we require there to be a canonical interpretation $\mathcal{J}$ over $(X \uplus V \uplus Y)$ which extends $\mathcal{I}$ with (S, F), and we require $\mathcal{J}$ to satisfy (T, B).

Motik *et al.* [13] use a similar notion of constraint satisfaction, although they consider all minimal $\mathcal{J}$, rather than a canonical $\mathcal{J}$, with respect to subset order on Herbrand models of Skolemized formulae. As they note, Skolemization has an impact on the notion of equivalence of TBoxes, for example $(c \sqsubseteq \exists r \,.\, d)$ is not equivalent to $(c \sqsubseteq \exists r \,.\, d, c \sqsubseteq \exists r \,.\, d)$ because they Skolemize differently (each existential quantifier introduces a new Skolem function, which may be interpreted differently). We avoid Skolemization by considering *initial* interpretations (relative to homomorphisms between interpretations) rather than *minimal* interpretations (relative to subset order).

Tao *et al.* [17] also consider minimal models, with respect to a partial order $\prec_{=}$ which preserves concept membership, role membership and equality of named individuals. They avoid Skolemization by an alternate semantics, where quantification only ranges over named individuals.

A *homomorphism* between interpretations $\mathcal{I}$ and $\mathcal{J}$ over X is a function $h : \Delta^{\mathcal{I}} \to \Delta^{\mathcal{J}}$ such that, for all x, i and j:

$$h(x^{\mathcal{I}}) = x^{\mathcal{J}} \qquad (i \in c^{\mathcal{I}}) \Rightarrow (h(i) \in c^{\mathcal{J}}) \qquad ((i,j) \in r^{\mathcal{I}}) \Rightarrow ((h(i), h(j)) \in r^{\mathcal{J}})$$

We will write $I \lesssim \mathcal{J}$ whenever there is a homomorphism from $\mathcal{I}$ to $\mathcal{J}$. Consider an interpretation $\mathcal{I}$, and a family of interpretations $\mathcal{J}_i$ with a chosen family of homomorphisms $h_i : \mathcal{I} \to \mathcal{J}_i$. An *initial* $\mathcal{J}_i$ is one with a unique family of homomorphisms: $g_j : \mathcal{J}_i \to \mathcal{J}_j$ such that $g_j \circ h_i = h_j$. Note that initial interpretations do not always exist, but that when they do they are unique up to isomorphism.

Definition 1. *For any interpretation $\mathcal{I}$ over X and ABox F over $Z \supseteq X$, let $\mathcal{I} \oplus (S, F)$ be the initial interpretation $\mathcal{J}$ over Z such that $\mathcal{I} \lesssim \mathcal{J}$ and $\mathcal{J} \vDash S, F$.*

Note that $\mathcal{I} \oplus (S, F)$ does not always exist, as S may contain existentials or disjunctions which do not have canonical witnesses. For example there is no initial extension of $\emptyset$ by:

$$\mathsf{Bool} \sqsubseteq \mathsf{True} \sqcup \mathsf{False} \qquad \mathsf{True} \sqcup \mathsf{False} \sqsubseteq \mathsf{Bool} \qquad \mathsf{Bool}(x)$$

since there are two incomparable extensions, one with $\mathsf{True}(x)$ and one with $\mathsf{False}(x)$. However, there is a syntactic restriction which guarantees the existence of initial interpretations. Let S be *minimizable* whenever any axiom $C \sqsubseteq D$ has C built from atoms, $\bot$, $\top$, $\sqcup$, $\sqcap$ and $\exists$, and D built from atoms, $\top$, $\sqcap$, $\forall$ and $\leq$.

Proposition 1. *If S is minimizable, then $\mathcal{I} \oplus (S, F)$ exists.*

4 Integrity constraints

Having defined initiality, we can now define constraint satisfaction. This is a variant of Motik *et al.*'s definition: rather than considering all minimal interpretations, we require a canonical initial interpretation to exist, and for it to satisfy the integrity constraints.

Definition 2. *For ABoxes A over X, B over Y and F over $(X \uplus V \uplus Y)$, define $F : A \Rightarrow B$ whenever, for any interpretation $\mathcal{I}$ over X such that $\mathcal{I} \vDash S, T, A$, we have $I \oplus (S, F) \vDash T, B$.*

For example, in the example from Section 1 we have that in any $\mathcal{I}$ which satisfies the ambient TBoxes and $\mathsf{PersonalHomePage}(\texttt{alice:})$, there must be some i such that $(\texttt{alice:}^{\mathcal{I}}, i) \in \mathsf{primaryTopic}^{\mathcal{I}}$, so we can pick fresh j and k and define $\mathcal{J}$ as the smallest extension of $\mathcal{I}$ where:

$$\texttt{bob:}^{\mathcal{J}} = j \qquad \texttt{bob:me}^{\mathcal{J}} = k \qquad \texttt{\_:anon}^{\mathcal{J}} = \texttt{alice:}^{\mathcal{I}}$$
$$j \in \mathsf{PersonalHomePage}^{\mathcal{J}} \qquad j \in \mathsf{Document}^{\mathcal{J}} \qquad k \in \mathsf{Person}^{\mathcal{J}}$$
$$(j,k) \in \mathsf{primaryTopic}^{\mathcal{J}} \qquad (k,j) \in \mathsf{homepage}^{\mathcal{J}} \qquad (k,i) \in \mathsf{knows}^{\mathcal{J}}$$

and so we have:

$$\begin{pmatrix} \mathsf{PersonalHomePage}(\texttt{bob:}), \\ \mathsf{Person}(\texttt{bob:me}), \\ \mathsf{primaryTopic}(\texttt{bob:}, \texttt{bob:me}), \\ \mathsf{knows}(\texttt{bob:me}, \texttt{\_:anon}), \\ \mathsf{homepage}(\texttt{\_:anon}, \texttt{alice:}) \end{pmatrix} : (\mathsf{PersonalHomePage}(\texttt{alice:})) \Rightarrow \begin{pmatrix} \mathsf{PersonalHomePage}(\texttt{bob:}), \\ \mathsf{Person}(\texttt{bob:me}) \end{pmatrix}$$

5 Symmetric monoidal category

Having defined our notion of integrity constraints for Linked Data, we give our main result, which is that ABoxes with integrity constraints form a symmetric monoidal category, and hence (as shown by Joyal and Street [10] and surveyed, for example, by Selinger [16]) can be modeled formally by directed acyclic graphs.

A symmetric monoidal category $\mathbf{C}$ consists of:

- A collection $\mathbf{Obj}(\mathbf{C})$ of *objects*, including:
  - a chosen object I, and
  - for each pair of objects A and B, an object $A \otimes B$.
- For each pair of objects, A and B, a collection of *morphisms* $\mathbf{C}[A, B]$, including (where we write $f : A \to B$ whenever f is in $\mathbf{C}[A, B]$):
  - for each $f : A \to B$ and $g : B \to C$, a morphism $(f; g) : A \to C$,
  - for each $f : A \to C$ and $g : B \to D$, a morphism $(f \otimes g) : (A \otimes B) \to (C \otimes D)$, and
  - chosen families of morphisms:

$$\begin{array}{rr} 1_A : A \to A & \sigma_{AB} : (A \otimes B) \to (B \otimes A) \\ \alpha_{ABC} : ((A \otimes B) \otimes C) \to (A \otimes (B \otimes C)) & \lambda_A : (A \otimes I) \to A \\ \alpha^{-1}_{ABC} : (A \otimes (B \otimes C)) \to ((A \otimes B) \otimes C) & \lambda^{-1}_A : A \to (A \otimes I) \end{array}$$

  satisfying certain equations (see, for example Mac Lane [12] for details).

The objects of our symmetric monoidal category **ABox** will be ABoxes, which we will think of as interfaces.

- $\mathbf{Obj}(\mathbf{ABox})$ is the collection of all ABoxes.
- The chosen object I is the empty ABox.
- Given two ABoxes A over X and B over Y, the object $(A \otimes B)$ is the ABox (A, B) over $(X \uplus Y)$.

The morphisms of the category **ABox** will also be ABoxes, this time thought of as datasets satisfying integrity constraints.

- $\mathbf{ABox}[A, B]$ is the collection of all ABoxes F such that $F : A \Rightarrow B$.
- Given two ABoxes F over $(X \uplus V \uplus Y)$ and G over $(Y \uplus W \uplus Z)$, the morphism $(F; G)$ is the ABox (F, G) over $(X \uplus (V \uplus Y \uplus W) \uplus Z)$.
- Given two ABoxes F_1 over $(X_1 \uplus V_1 \uplus Y_1)$ and F_2 over $(X_2 \uplus V_2 \uplus Y_2)$, the morphism $(F_1 \otimes F_2)$ is the ABox (F_1, F_2) over $((X_1 \uplus X_2) \uplus (V_1 \uplus V_2) \uplus (Y_1 \uplus Y_2))$.

To verify that this definition is well-formed, we have to verify that checking integrity constraints is compositional, that is we only have to check integrity locally, and know it is preserved by composition and tensor.

Proposition 2.

1. *If* $F : A \Rightarrow B$ *and* $G : B \Rightarrow C$*, then* $(F;G) : A \Rightarrow C$*.*
2. *If* $F_1 : A_1 \Rightarrow B_1$ *and* $F_2 : A_2 \Rightarrow B_2$*, then* $(F_1 \otimes F_2) : (A_1 \otimes A_2) \Rightarrow (B_1 \otimes B_2)$*.*

Note that the composition $(F;G)$ may introduce bnodes, since the intermediate names which are exported by F and imported by G become bnodes (indeed, this is why bnodes are present in this model). For example:

$$\begin{aligned}&(\mathsf{knows}(\texttt{alice:me}, \texttt{bob:me})); (\mathsf{knows}(\texttt{bob:me}; \texttt{charlie:me}))\\ &\quad \equiv (\mathsf{knows}(\texttt{alice:me}, \texttt{\_:anon}), \mathsf{knows}(\texttt{\_:anon}, \texttt{charlie:me}))\end{aligned}$$

As well as composition of ABoxes, we have to provide the "wiring" combinators for identity, symmetry, unit and associativity. These are all constructed in the same way: given any function $f : Y \to X$ on finite sets, we define the ABox $\mathsf{wiring}(f)$ over $(X \uplus Y)$ as containing $f(y) \approx y$ for each $y \in Y$. We can then show that $\mathsf{wiring}(f)$ respects renaming of ABoxes. Given any ABox A over Y, let $f[A]$ be the ABox over X given by replacing any individual y in A by $f(y)$.

Proposition 3. *If* $f : Y \to X$ *and* $B \subseteq f[A]$*, then* $\mathsf{wiring}(f) : A \Rightarrow B$*.*

This suffices to define the combinators of a symmetric monoidal category, for example $1_A : A \Rightarrow A$ is given by wiring the identity function.

Finally, we have to prove the equations of a symmetric monoidal category. These equations are *not* true up to syntactic equality of ABoxes, due to introduction of bnodes, for example a counter-example to $1; F = F$ is:

$$\begin{aligned}&(\texttt{alice:me} \approx \texttt{alice:me}'); (\mathsf{knows}(\texttt{alice:me}', \texttt{bob:me}))\\ &\quad \equiv (\texttt{alice:me} \approx \texttt{\_:anon}, \mathsf{knows}(\texttt{\_:anon}, \texttt{bob:me}))\\ &\quad \neq (\mathsf{knows}(\texttt{alice:me}, \texttt{bob:me}))\end{aligned}$$

The equations *are* true when we consider ABoxes up to equivalence (in the presence of S, T and A), that is:

$$F \equiv G : A \Rightarrow B \text{ whenever } S, T, A, F \vDash G \text{ and } S, T, A, G \vDash F$$

We therefore consider the morphisms of **ABox** up to equivalence, which requires us to show that composition and tensor respect equivalence:

Proposition 4.

1. *If* $F \equiv F' : A \Rightarrow B$ *and* $G \equiv G' : B \Rightarrow C$ *then* $(F;G) \equiv (F';G') : A \Rightarrow C$*.*
2. *If* $F_1 \equiv F_1' : A_1 \Rightarrow B_1$ *and* $F_2 \equiv F_2' : A_2 \Rightarrow B_2$
   then $(F_1 \otimes F_2) \equiv (F_1' \otimes F_2') : (A_1 \otimes A_2) \Rightarrow (B_1 \otimes B_2)$*.*

Fig. 3. Example of Agda proof mechanization

The proofs that ABoxes satisfy the equations of a symmetric monoidal category are then direct. The coherence properties (which only involve compositions of wiring morphisms) follow because wiring respects composition and tensor:

$$\mathsf{wiring}(f);\mathsf{wiring}(g) \equiv \mathsf{wiring}(f \circ g) \qquad \mathsf{wiring}(f) \otimes \mathsf{wiring}(g) \equiv \mathsf{wiring}(f \uplus g)$$

Theorem 1. ABox *forms a symmetric monoidal category.*

The proof of this theorem, including the definitions it relies on, is approximately 3,000 lines of Agda code [9]. An example lemma is shown in Figure 3.

6 Conclusions and further work

We have presented the first treatment of integrity constraints for Linked Data which makes use of a partition between local entities, for which a dataset is authoritative, and imported entities, where complete information is not known. We have given the first categorical presentation of datasets, and as a consequence, we have the first formal treatment of acyclic dependency graphs.

There are open questions raised by this model, of which the most important is its algorithmic properties: is integrity constraint satisfaction decidable, and if so, what is its complexity, and can it be reduced to existing decision problems?

Our model only treats acyclic dependency graphs, via symmetric monoidal categories. A categorical treatment of cyclic graphs uses *traced* monoidal categories (introduced by Joyal, Street and Verity [11], and discussed by Selinger [16]). Cyclic graphs require the existence of fixed points which unfortunately do not respect integrity constraint satisfaction, for example the fixed point of the identity morphism is equivalent to an empty dataset, which will not satisfy existential

or disjunctive integrity constraints. The situation is similar to that of complete metric spaces: not all functions have fixed points, but contraction maps do.

Our model assumes the existence of ambient TBoxes S and T, which must be agreed upon by all datasets. This requirement is quite strong, and the model would be improved by allowing authoritative ontologies as well as datasets. This is related to the notion of *modularity* of ontologies [7].

The mechanized proofs of our model [9] are given in Agda [1], which as well as a proof assistant is a programming language which compiles to Haskell [3]. We hope to extend our proofs to a Semantic Web library, which will support the development of provably correct programs to process Linked Data.

References

1. The Agda programming language. `http://wiki.portal.chalmers.se/agda/`
2. The friend of a friend (FOAF) project, `http://www.foaf-project.org/`
3. The Haskell programming language. `http://haskell.org/`
4. Beckett, D., Berners-Lee, T.: Turtle - terse RDF triple language (2008), `http://www.w3.org/TeamSubmission/turtle/`
5. Berners-Lee, T.: Linked data (2006), `http://www.w3.org/DesignIssues/LinkedData.html`
6. Cyganiak, R., Jentzsch, A.: Linking open data cloud diagram, `http://lod-cloud.net/`
7. Grau, B.C., Horrocks, I., Kazakov, Y., Sattler, U.: Modular reuse of ontologies: Theory and practice. J. Artificial Intelligence Research 31, 273–318 (2008)
8. Jacobs, I., Walsh, N.: Architecture of the World Wide Web, volume one. W3C Recommendation (2004), `http://www.w3.org/TR/webarch/`
9. Jeffrey, A.S.A.: Agda libraries for the semantic web. `https://github.com/agda/agda-web-semantic/` (2011)
10. Joyal, A., Street, R.: The geometry of tensor calculus I. Advances in Mathematics 88(1), 55–112 (1991)
11. Joyal, A., Street, R., Verity, D.: Traced monoidal categories. Math. Proc. Cambridge Phil. Soc. 3, 447–468 (1996)
12. Mac Lane, S.: Categories for the Working Mathematician. Springer, 2nd edn. (1998)
13. Motik, B., Horrocks, I., Sattler, U.: Bridging the gap between OWL and relational databases. J. Web Semantics 7(2), 74–89 (2009)
14. Motik, B., Rosati, R.: Reconciling Description Logics and Rules. J. ACM 57(5), 1–62 (2010)
15. Reiter, R.: What should a database know? J. Log. Program. 14, 127–153 (1992)
16. Selinger, P.: A survey of graphical languages for monoidal categories. In: Coecke, B. (ed.) New Structures for Physics, Lecture Notes in Physics, vol. 813, chap. 4, pp. 289–356. Springer (2011)
17. Tao, J., Sirin, E., Bao, J., McGuinness, D.L.: Extending OWL with integrity constraints. In: Proc. Workshop on Description Logics. pp. 137–148 (2010)