

Modeling the Mediated Schema Constraints

Marco A. Casanova¹, Tanara Lauschner¹, Luiz André P. Paes Leme¹,
Karin K. Breitman¹, Antonio L. Furtado¹, Vânia M.P. Vidal²

¹*Department of Informatics – Pontifical Catholic University of Rio de Janeiro
Rio de Janeiro, RJ – Brazil*

{casanova, tanara, lleme, karin, furtado}@inf.puc-rio.br

²*Department of Computing, Federal University of Ceará
Fortaleza, CE – Brazil
vvidal@lia.ufc.br*

Abstract

In this paper, we address the problem of modeling the constraints of a mediated schema. We argue that, from the point of view of an application that processes the results of queries defined over the mediated schema, the constraints should be modeled as the greatest lower bound of the constraints of the export schemas, after appropriate translation to a common vocabulary. This assures that the application will correctly interpret query results.

1. Introduction

A mediated environment consists of: (1) a mediated schema M ; (2) for each $i \in [1, n]$, an export schema E_i , an import schema I_i , and a local mapping γ_i ; (3) a mediated mapping μ . The mediated schema M provides a common description of the data sources that participate in the mediated environment. The export schema E_i indicates how the data exported by the i^{th} data source is organized, I_i describes how such data is reorganized when imported to the mediated environment (hence the term *import schema*), γ_i defines the concepts of I_i in terms of the concepts of E_i , and μ defines the concepts of M in terms of the concepts of the import schemas. We may also understand M as defining a common vocabulary, I_i as defining the subset of such vocabulary that the i^{th} data source supports, and γ_i as defining how to interpret such subset in terms of the vocabulary of E_i .

We introduce the notion of import schema mostly to break the definition of the mappings into two stages: the definition of the local mappings and the definition of the mediated mapping. Furthermore, we restrict the mediated mapping so that it defines the concepts of M as unions of concepts from the import schemas. This

restriction is justified in the context of the Web, where data sources are completely independent, in the sense that the mediated environment has little control when a data source joins or leaves the environment. Hence, it should be a simple process to add or remove a data source, without propagating changes beyond the mediated mapping. This restriction also simplifies translating queries and updates over the mediated schema to queries and updates over the import schemas.

There are two central questions to be addressed: the derivation of the constraints of the import schemas from the constraints of the export schemas, and the derivation of the constraints of the mediated schema from the constraints of the import schemas.

Consider first the derivation of the constraints of an import schema. Let E_i be an export schema, I_i be the corresponding import schema, and γ_i be the corresponding local mapping. We require that the constraints of I_i be *correct* in the sense that, for any state s_E of E_i that satisfies the constraints of E_i , if γ_i maps s_E into a state s_I of I_i , then s_I must satisfy the constraints of I_i . Intuitively, if correct, the constraints of I_i express the semantics of the data exported by E_i , after applying the transformation to the common vocabulary of the mediated schema, as expressed by γ_i . We refer the reader to [10] for a procedure to generate correct constraints for I_i from the constraints of E_i .

The central theme of this paper is the derivation of the constraints of the mediated schema. We argue that one should maintain a set of constraints that we call the *query constraints*, C_Q .

For $i=1,2$, let Φ_i be a set of constraints and $Th(\Phi_i)$ denote the set of constraints derived from Φ_i . We define the *greatest lower bound* (g.l.b.) as

$$\Phi_1 \triangle \Phi_2 = Th(\Phi_1) \cap Th(\Phi_2)$$

Note that any constraint in the greatest lower bound of Φ_1 and Φ_2 can be derived from Φ_1 and, separately, from Φ_2 . We say that Φ_1 and Φ_2 are tautologically equivalent iff, for any constraint σ , we have that σ can be derived from Φ_1 iff σ can be derived from Φ_2 .

The purpose of the query constraints is to convey to an application that uses the mediated schema the common semantics of the data returned from the data sources, so that the application has enough information to correctly process query results. As such, we claim that the query constraints should be modeled to be tautologically equivalent to the greatest lower bound of the sets of constraints of the import schemas. This implies that the data returned by queries defined over the mediated schema will always satisfy the query constraints. Indeed, the data returned by each data source will satisfy the constraints of each import schema (which are correct by construction) and, consequently, the query constraints of the mediated schema (which are modeled as the greatest lower bound of the constraints of the import schemas).

For example, consider a virtual store mediating access to a collection of online booksellers. Assume that some booksellers admit books with zero or more author names, whereas others admit books with one or more author names. Then, the query constraints must allow books with zero or more author names to account for data returned by any bookseller. Hence, an application has to test if an author name exists or not when processing a book. Albeit simple, this example illustrates how applications should use query constraints.

The results reported here are part of a larger effort to create mediators for data sources available on the Web, using instance-based schema matching algorithms [10]. Research in schema matching, as well as in ontology matching [8], tends to concentrate on vocabulary matching techniques, ignoring the question of constraint definition. Calvanese et al. [4] introduce a Description Logics framework, similar to that in Section 2, to address data integration and query answering. Atzeni et al. [1] cover the traditional problem of rewriting a schema from one model to another, but they do not touch on the more complex problem of generating a set of constraints that generalizes a pair of sets of constraints from different schemas, which we address in Sections 4 and 5. Chiticariu et al. [6] and Cali et al. [5] use integrity constraints to expand queries, whereas our focus is on the synthesis of the constraints of the mediated schema from the constraints of the export schemas. Curino et al. [7] describe a software tool to support schema evolution that uses mapping invertibility. Fagin et al. [9] study mapping invertibility in the context of

source-to-target tuple generating dependencies and formalize the notion of quasi-inverse. By contrast, we show in Section 3 how to generate the best possible set of constraints without computing inverse mappings.

This paper is organized as follows. Section 2 introduces an expressive family of conceptual schemas and a family of mappings. With the help of an example, Sections 3 and 4 discuss what it means to maintain the set of query constraints of a mediated schema. Finally, Section 5 contains the conclusions.

2. Basic definitions

2.1. A brief review of Description Logics

We adopt a family of attributive languages [2] defined as follows. A language \mathcal{L} in the family is characterized by an *alphabet* \mathcal{A} , consisting of a set of *atomic concepts*, a set of *atomic roles*, a set of *constants*, the *universal* and the *bottom concepts*, denoted by \top and \perp , respectively, and the *universal* and the *bottom roles*, also denoted by \top and \perp , respectively.

The set of *role descriptions* of \mathcal{L} is inductively defined as

- An atomic role and the universal and bottom roles are role descriptions
- If p and q are role descriptions, then the following expressions are role descriptions
 - p^- (the inverse of p)
 - $p \circ q$ (the composition of p and q)

The set of *concept descriptions* of \mathcal{L} is inductively defined as

- An atomic concept and the universal and bottom concepts are concept descriptions
- If a_1, \dots, a_n are constants, then $\{a_1, \dots, a_n\}$ is a concept description
- If e and f are concept descriptions, p is a role description and n a non-negative integer, then the following expressions are concept descriptions
 - $\neg e$ (negation)
 - $e \sqcap f$ (intersection)
 - $e \sqcup f$ (union)
 - $\exists p$ (restricted existential quantification)
 - $\exists p.e$ (full existential quantification)
 - $\forall p.e$ (value restriction)
 - $(\leq n p)$ (at-most restriction)
 - $(\geq n p)$ (at-least restriction)

An *interpretation* s for the symbols in the alphabet \mathcal{A} consists of a nonempty set Δ^s , the *domain* of s , whose elements are called *individuals*, and an *interpretation function*, also denoted s , where:

- $s(\top) = \Delta^s$, when \top denotes the universal concept
- $s(\perp) = \emptyset$, when \perp denotes the bottom concept
- $s(A) \subseteq \Delta^s$, for each atomic concept A
- $s(\top) = \Delta^s \times \Delta^s$, when \top denotes the universal role
- $s(\perp) = \emptyset$, when \perp denotes the bottom role
- $s(P) \subseteq \Delta^s \times \Delta^s$, for each atomic role P
- $s(a) \in \Delta^s$, for each constant a , such that distinct constants denote distinct individuals (the *uniqueness assumption*)

Function s is extended to role and concept descriptions of \mathcal{L} as follows:

- $s(p \neg) = s(p)^\neg$ (the inverse of $s(p)$)
- $s(p \circ q) = s(p) \circ s(q)$
(the composition of $s(p)$ with $s(q)$)
- $s(\{a_1, \dots, a_n\}) = \{s(a_1), \dots, s(a_n)\}$
- $s(\neg e) = \Delta^s - s(e)$
(the complement of $s(e)$ w.r.t. Δ^s)
- $s(e \sqcap f) = s(e) \cap s(f)$
(the intersection of $s(e)$ and $s(f)$)
- $s(e \sqcup f) = s(e) \cup s(f)$
(the union of $s(e)$ and $s(f)$)
- $s(\exists p) = \{I \in \Delta^s / (\exists J \in \Delta^s)((I, J) \in s(p))\}$
(the set of individuals that $s(p)$ relates to some individual)
- $s(\exists p.e) = \{I \in \Delta^s / (\exists J \in \Delta^s)((I, J) \in s(p) \wedge J \in s(e))\}$
(the set of individuals that $s(p)$ relates to some individual in $s(e)$)
- $s(\forall p.e) = \{I \in \Delta^s / (\forall J \in \Delta^s)((I, J) \in s(p) \Rightarrow J \in s(e))\}$
(the set of individuals I such that, if $s(p)$ relates I to an individual J , then J is in $s(e)$)
- $s(\geq n p) = \{I \in \Delta^s / |\{J \in \Delta^s / (I, J) \in s(p)\}| \geq n\}$
(the set of individuals that $s(p)$ relates to at least n distinct individuals)
- $s(\leq n p) = \{I \in \Delta^s / |\{J \in \Delta^s / (I, J) \in s(p)\}| \leq n\}$
(the set of individuals that $s(p)$ relates to at most n distinct individuals)

A *formula* of \mathcal{L} is an expression of the form $u \sqsubseteq v$, called an *inclusion*, or of the form $u \equiv v$, called an *equivalence*, where u and v are both concept descriptions or they are both role descriptions of \mathcal{L} . A *definition* is an equivalence of the form $T \equiv u$, where T is an atomic concept and u is a concept description, or T is an atomic role and u is a role description. An interpretation s for \mathcal{L} *satisfies* $u \sqsubseteq v$ iff $s(u) \subseteq s(v)$, and s *satisfies* $u \equiv v$ iff $s(u) = s(v)$.

We say that an inclusion of the form $u \sqsubseteq \neg v$ is an *exclusion*, which we abbreviate as $u \perp v$; we also say that u and $\neg v$ are *disjoint* or *mutually exclusive*. Hence, an interpretation s for \mathcal{L} *satisfies* $u \sqsubseteq \neg v$ iff $s(u) \cap \neg s(v) = \emptyset$.

In the rest of the paper, we will use the following notation:

- $s \models \sigma$ indicates that an interpretation s satisfies a formula σ
- $s \models \Sigma$ indicates that an interpretation s satisfies all formulas in a set of formulas Σ
- $\Sigma \models \sigma$ indicates that a set of formulas Σ *logically implies* a formula σ , that is, for any interpretation s , if $s \models \Sigma$, then $s \models \sigma$
- $\Sigma \models \Gamma$ indicates that a set of formulas Σ *logically implies* a set of formulas Γ , that is, for any interpretation s , if $s \models \Sigma$, then $s \models \Gamma$
- $Th(\Sigma)$ denotes the *theory induced* by Σ , which is the smallest set of formulas that contains Σ and is closed under logical implication
- Σ and Γ are *tautologically equivalent* iff $Th(\Sigma) = Th(\Gamma)$

We will also use concept and role descriptions over an alphabet \mathcal{A} which is the union of alphabets $\mathcal{A}_1, \dots, \mathcal{A}_n$ whose only symbols in common are \top and \perp . The syntax of concept and role descriptions remains the same. An interpretation s for \mathcal{A} is constructed from interpretations s_1, \dots, s_n for $\mathcal{A}_1, \dots, \mathcal{A}_n$ as follows:

- the domain of s is the union of the domains of s_1, \dots, s_n
- $s(x) = s_i(x)$ iff x is a symbol, other than \top and \perp , of \mathcal{A}_i (which is unique by assumption)
- $s(\top)$ and $s(\perp)$ are defined as usual

We also assume that (*Domain Disjointness Assumption*) Any pair of interpretations for \mathcal{A}_i and \mathcal{A}_j have disjoint domains, for each $i, j \in [1, n]$, with $i \neq j$.

2.2. Extralite schemas

An *extralite schema* is a pair $S = (\mathcal{A}, \Sigma)$ such that

- \mathcal{A} is an alphabet, called the *vocabulary* of S , whose atomic concepts and atomic roles are called the *classes* and *properties* of S , respectively
- Σ is a set of formulas, called the *constraints* of S , which must be of one the forms (where C and D are classes and P is a property of S)
 - *Domain Constraint*: $\exists P \sqsubseteq C$
(property P has domain C)
 - *Range Constraint*: $\exists P^\neg \sqsubseteq C$
(property P has range C)
 - *minCardinality constraint*: $C \sqsubseteq (\geq k P)$
(property P maps each individual in C to at least k distinct individuals)
 - *maxCardinality constraint*: $C \sqsubseteq (\leq k P)$

(property P maps each individual in C to at most k distinct individuals)

- *Subset Constraint:* $C \sqsubseteq D$
(class C is a subclass of class D)
- *Exclusion Constraint:* $C \perp D$
(class C is disjoint from class D)

We also say that an interpretation s for \mathcal{A} is a *consistent state* of S iff s satisfies all constraints in Σ .

We observe that constraints are very restricted formulas, but they capture enough semantics to model simple UML diagrams. They also cover some of the property restrictions and class axioms of OWL [3].

From this point on, we will use the terms *class*, *property* and *vocabulary* interchangeably with the terms *atomic concept*, *atomic role* and *alphabet*, respectively.

The domain and range constraints are collectively called *property constraints*, the minCardinality and maxCardinality constraints are called *cardinality constraints*, and the subset and exclusion constraints are called *class constraints*. Note that we allow more than one domain or range constraint per property, and more than one minCardinality or maxCardinality constraint for the same class and property pair.

Furthermore, we will use $D \sqsubseteq (=k P)$ as an abbreviation for a pair of constraints of the form $D \sqsubseteq (\geq k P)$ and $D \sqsubseteq (\leq k P)$. For later reference, we also observe that we may rewrite a domain constraint $\exists P \sqsubseteq C$ as $(\geq 1 P) \sqsubseteq C$, and a range constraint $\exists P^- \sqsubseteq C$ as $(\geq 1 P^-) \sqsubseteq C$, since $\exists P \equiv (\geq 1 P)$ and $\exists P^- \equiv (\geq 1 P^-)$.

Property constraints deserve some explanation. Let s be an interpretation for \mathcal{A} with domain Δ^s . Then, s satisfies a domain constraint $\exists P \sqsubseteq C$ iff, for any $a \in \Delta^s$, if $a \in s(\exists P)$ then $a \in s(C)$. But $a \in s(\exists P)$ iff there is $b \in \Delta^s$ such that $(a, b) \in s(P)$, which means that a is in the domain of $s(P)$. In other words, if a is in the domain of $s(P)$, then $a \in s(C)$.

Likewise, s satisfies a range constraint $\exists P^- \sqsubseteq C$ iff, for any $b \in \Delta^s$, if $b \in s(\exists P^-)$ then $b \in s(C)$. But $b \in s(\exists P^-)$ iff there is $a \in \Delta^s$ such that $(a, b) \in s(P)$, which means that b is in the range of $s(P)$. In other words, if b is in the range of $s(P)$, then $b \in s(C)$.

Finally, note that this formalization does not distinguish between object and datatype properties, in OWL terminology. The formal development does not capture this distinction since the notion of the domain of an interpretation does not separate individuals that denote class elements from individuals that correspond to datatype values. The distinction will be visible in the examples, however, where the range of an object property will be a class defined in the schema, whereas

the range of a datatype property will be a XML Schema type (i.e, a set of datatype values or *literals*).

Example 1: Figures 1(a), (c) and (e) show schemas for fragments of the Amazon, the eBay and the Barnes&Noble (BN) databases, using an informal notation. We use the namespace prefixes “a:”, “e:” and “b:” to refer to the vocabularies of the Amazon, the eBay and the BN schemas.

In Figure 1(a), for example, a:title is defined as a (datatype) property with domain a:Product and range string (an XML Schema data type), a:Book is declared as a subclass of a:Product, and a:pub is defined as an (object) property with domain a:Book and range a:Publ.

Figures 1(b), (d) and (f) formalize the constraints: the first column shows the property constraints; the second column, the cardinality constraints; and the third column, the class constraints. Just to help illustrate the discussion in Section 3, we assume that a:pub has minCardinality 2 and that a:name has minCardinality 3.

2.3. Mediated environment

A *mediated environment* contains a *mediated schema* $M=(\mathcal{A}, \Sigma_Q)$, a *mediated mapping* μ and, for each $k=1, \dots, n$, an *export schema* E_k , an *import schema* I_k and a *local mapping* γ_k . We also say that Σ_Q is the set of *query constraints* of M .

In what follows, assume that C_1, \dots, C_u are the classes and P_1, \dots, P_v are the properties of the vocabulary of the mediated schema M .

We restrict the import schemas as follows:

- for $k=1, \dots, n$, the vocabulary of I_k is a subset of the vocabulary of M

In the definitions that follow, we do not adopt namespace prefixes, as in the examples, but a more abstract notation to distinguish the occurrence of a symbol in the vocabulary of M from the occurrence of the same symbol in the vocabulary of I_k . For each class C_i (or property P_j) in the vocabulary of M , we denote the occurrence of C_i (or P_j) in the vocabulary of I_k by C_i^k (or P_j^k); we also say that C_i^k (or P_j^k) *matches* C_i (or P_j).

The mediated mapping μ defines the classes and properties of M as unions of classes and properties of the import schemas. More precisely, we restrict the mediated mapping μ as follows:

- for each $i=1, \dots, u$, μ contains a definition of the form

$$C_i \equiv e_i^1 \sqcup \dots \sqcup e_i^n$$

where e_i^k is the class C_i^k of I_k that matches C_i , if it exists, or the bottom concept \perp , otherwise, for each $k=1, \dots, n$

- for each $j=1, \dots, v$, μ contains a definition of the form

$$P_j \equiv p_j^1 \sqcup \dots \sqcup p_j^n$$

where p_j^k is the property P_j^k of I_k that matches P_j , if it exists, or the bottom role \perp , for each $k=1, \dots, n$

Note that the definition of C_i (or P_j) has exactly one concept description (or role description) from each import schema, even if the import schema has no class (or property) that matches C_i (or P_j), in which case we use \perp .

For each $k=1, \dots, n$, the local mapping γ_k defines the classes and properties of I_k in the terms of the vocabulary of the export schema E_k . We restrict the local mapping γ_k as follows:

- for each class C_i^k of I_k , γ_k contains a definition of the form

$$C_i^k \equiv \rho_i^k$$

where ρ_i^k is a concept description over the vocabulary of E

- for each property P_j^k of I_k , γ_k contains a definition of the form

$$P_j^k \equiv \pi_j^k$$

where π_j^k is a role description over the vocabulary of E_k

We introduce $\bar{\gamma}_k$ as the function from states of E_k into states of I_k such that, for each state s of E_k , $\bar{\gamma}_k(s) = r$ iff

- $r(C_i^k) = s(\rho_i^k)$, if $C_i^k \equiv \rho_i^k$ defines class C_i^k in γ_k
- $r(P_j^k) = s(\pi_j^k)$, if $P_j^k \equiv \pi_j^k$ defines property P_j^k in γ_k

We say that $\bar{\gamma}_k$ is the *function induced by* the local mapping γ_k .

Likewise, we introduce $\bar{\mu}$ as the function from states of E_1, \dots, E_n into states of M such that, for states s_1, \dots, s_n

of E_1, \dots, E_n , $\bar{\gamma}(s_1, \dots, s_n) = r$ iff, for $i=1, \dots, u$ and $j=1, \dots, v$, we have

- $r(C_i) = s_1(e_i^1) \cup \dots \cup s_n(e_i^n)$,
if $C_i \equiv e_i^1 \sqcup \dots \sqcup e_i^n$ defines C_i in μ
- $r(P_j) = s_1(p_j^1) \cup \dots \cup s_n(p_j^n)$,
if $P_j \equiv p_j^1 \sqcup \dots \sqcup p_j^n$ defines P_j in μ

We say that $\bar{\mu}$ is the *function induced by* the mediated mapping μ and the local mapping $\gamma_1, \dots, \gamma_n$.

Example 2: Figure 2 describes a mediated environment, with:

- the mediated schema `Sales`, whose vocabulary is shown in Figure 2(a)
- the Amazon, eBay and BN schemas, shown in Figure 1, as export schemas
- the import schema for the Amazon export schema (not shown in Figure 2), with the same classes and properties as `Sales`, but prefixed with “ai:”
- the import schema for the eBay export schema (not shown in Figure 2), with the same classes and properties as `Sales`, but prefixed with “ei:”
- the import schema for the BN export schema (not shown in Figure 2), with the same classes and properties as `Sales`, but prefixed with “bi:”
- the mediated mapping shown in Figure 2(b)
- the local mapping, shown in Figure 2(c), defining the classes and properties of the Amazon import schema in terms of its export schema; in particular, `ai:pub` is defined as the composition of `a:pub` with `a:name`
- the local mapping, shown in Figure 2(d), defining the classes and properties of the eBay import schema in terms of its export schema; in particular, `ei:Music` and `ei:Book` are defined as concept descriptions involving `e:Product`.
- the local mapping, shown in Figure 2(e), defining the classes and properties of the BN import schema in terms of its export schema.

a:Product a:title range string a:price range decimal a:currency range string	a:Book is-a a:Product a:Music is-a a:Product a:Video is-a a:Product a:PC-HW is-a a:Product
a:Book a:isbn range string a:author range string a:pub range a:Publ	a:Book, a:Music, a:Video, a:PC-HW are mutually disjoint
a:Publ a:name range string a:address range string	

Fig. 1(a). Informal definition of the Amazon schema.

\exists a:title \sqsubseteq a:Product \exists a:title ⁻ \sqsubseteq string ...	a:Product \sqsubseteq (= 1 a:title) a:Product \sqsubseteq (= 1 a:price) a:Product \sqsubseteq (= 1 a:currency) a:Book \sqsubseteq (= 1 a:isbn) a:Book \sqsubseteq (\geq 2 a:pub) a:Publ \sqsubseteq (\geq 3 a:name) a:Publ \sqsubseteq (= 1 a:address)	a:Book \sqsubseteq a:Product a:Music \sqsubseteq a:Product a:Video \sqsubseteq a:Product a:PC-HW \sqsubseteq a:Product a:Book a:Music a:Book a:Video a:Book a:PC-HW a:Music a:Video a:Music a:PC-HW a:Video a:PC-HW
\exists a:pub \sqsubseteq a:Book \exists a:pub ⁻ \sqsubseteq a:Publ ...		
\exists a:name \sqsubseteq a:Publ \exists a:name ⁻ \sqsubseteq string		

Fig. 1(b). Formal definition of (some of) the constraints of the Amazon schema.

e:Seller e:name range string	e:Product e:type range string e:ean range integer e:title range string e:author range string e:edition range integer e:year range integer e:pub range string
e:Offer e:qty range integer e:price range double e:currency range string e:seller range e:Seller e:product range e:Product	

Fig. 1(c). Informal definition of the eBay schema.

\exists e:name \sqsubseteq e:Seller \exists e:name ⁻ \sqsubseteq string ...	e:Seller \sqsubseteq (= 1 e:name) e:Offer \sqsubseteq (= 1 e:qty) e:Offer \sqsubseteq (= 1 e:price) e:Offer \sqsubseteq (= 1 e:currency) e:Offer \sqsubseteq (= 1 e:seller) e:Offer \sqsubseteq (\geq 1 e:product) e:Product \sqsubseteq (= 1 e:type) e:Product \sqsubseteq (= 1 e:ean) e:Product \sqsubseteq (= 1 e:title) e:Product \sqsubseteq (\geq 1 e:author) e:Product \sqsubseteq (= 1 e:edition) e:Product \sqsubseteq (= 1 e:year) e:Product \sqsubseteq (\geq 1 e:pub)	(no class constraints)
\exists e:seller \sqsubseteq e:Offer \exists e:seller ⁻ \sqsubseteq e:Seller \exists e:product \sqsubseteq e:Offer \exists e:product ⁻ \sqsubseteq e:Product ...		
\exists e:title \sqsubseteq e:Product \exists e:title ⁻ \sqsubseteq string \exists e:pub \sqsubseteq e:Product \exists e:pub ⁻ \sqsubseteq string		

Fig. 1(d). Formal definition of (some of) the constraints of the eBay schema.

b:Product b:title range string	b:CultProd is-a b:Product b:Music is-a b:CultProd b:Book is-a b:CultProd
b:Book b:pub range string	

Fig. 1(e) Informal definition of the Barnes&Noble (BN) schema.

\exists b:title \sqsubseteq b:Product \exists b:title ⁻ \sqsubseteq string \exists b:pub \sqsubseteq b:Book \exists b:pub ⁻ \sqsubseteq string	b:Book \sqsubseteq (\geq 2 b:pub)	b:Book \sqsubseteq b:CultProd b:Music \sqsubseteq b:CultProd b:CultProd \sqsubseteq b:Product
---	--	---

Fig. 1(f). Formal definition of the constraints of the Barnes&Noble (BN) schema.

Classes: s:Product, s:Music, s:Book	Properties: s:title, s:pub
-------------------------------------	----------------------------

Fig. 2(a). The Vocabulary of the Sales mediated schema.

s:Product \equiv ai:Product \sqcup ei:Product \sqcup bi:Product s:Music \equiv ai:Music \sqcup ei:Music \sqcup bi:Music s:Book \equiv ai:Book \sqcup ei:Book \sqcup bi:Book	s:title \equiv ai:title \sqcup ei:title \sqcup bi:title s:pub \equiv ai:pub \sqcup ei:pub \sqcup bi:pub
---	--

Fig. 2(b). Mediated schema mapping.

ai:Product \equiv a:Product ai:Music \equiv a:Music ai:Book \equiv a:Book	ai:title \equiv a:title ai:pub \equiv a:pub \circ a:name
---	---

Fig. 2(c). Local schema mappings from the Amazon export schema to its import schema.

ei:Product \equiv e:Product ei:Music \equiv e:Product \sqcap \exists e:type.{'music'} ei:Book \equiv e:Product \sqcap \exists e:type.{'book'}	ei:title \equiv e:title ei:pub \equiv e:pub
---	--

Fig. 2(d). Local schema mappings from the eBay export schema to its import schema.

bi:Product \equiv b:Product bi:Music \equiv b:Music bi:Book \equiv b:Book	bi:title \equiv b:title bi:pub \equiv b:pub
---	--

Fig. 2(e). Local schema mappings from the BN export schema to its import schema.

\exists ai:title \sqsubseteq ai:Product \exists ai:title ⁻ \sqsubseteq string \exists ai:pub \sqsubseteq ai:Book \exists ai:pub ⁻ \sqsubseteq string	ai:Product \sqsubseteq (= 1 ai:title) ai:Book \sqsubseteq (\geq 6 ai:pub)	ai:Book \sqsubseteq ai:Product ai:Music \sqsubseteq ai:Product ai:Book ai:Music
---	---	---

Fig. 3(a). Constraints of the Amazon Import Schema.

\exists ei:title \sqsubseteq ei:Product \exists ei:title ⁻ \sqsubseteq string \exists ei:pub \sqsubseteq ei:Product \exists ei:pub ⁻ \sqsubseteq string	ei:Product \sqsubseteq (= 1 ei:title) ei:Product \sqsubseteq (\geq 1 ei:pub)	ei:Book \sqsubseteq ei:Product ei:Music \sqsubseteq ei:Product ei:Book ei:Music
--	--	---

Fig. 3(b). Constraints of the eBay Import Schema.

\exists bi:title \sqsubseteq bi:Product \exists bi:title ⁻ \sqsubseteq string \exists bi:pub \sqsubseteq bi:Book \exists bi:pub ⁻ \sqsubseteq string	bi:Book \sqsubseteq (\geq 2 bi:pub)	bi:Book \sqsubseteq bi:Product bi:Music \sqsubseteq bi:Product
---	--	---

Fig. 3(c). Constraints of the BN Import Schema.

\exists s:title \sqsubseteq s:Product \exists s:title ⁻ \sqsubseteq string \exists s:pub \sqsubseteq s:Product \exists s:pub ⁻ \sqsubseteq string	s:Book \sqsubseteq (\geq 1 s:pub)	s:Book \sqsubseteq s:Product s:Music \sqsubseteq s:Product
--	--	---

Fig. 3(d). Query Constraints of the Sales Mediated Schema.

3. Modeling the import schema constraints

Recall that $Th(\Gamma)$ denotes the theory induced by a set of formulas Γ . Let \mathcal{U} be the set of all sets of constraints. Then, $(\mathcal{U}, \nabla, \Delta)$ is a lattice where, given any two sets of constraints, Γ_1 and Γ_2 , their greatest lower bound (g.l.b.) is $\Gamma_1 \Delta \Gamma_2 = Th(\Gamma_1) \cap Th(\Gamma_2)$ and their least upper bound (l.u.b.) is $\Gamma_1 \nabla \Gamma_2 = Th(\Gamma_1) \cup Th(\Gamma_2)$. Note that $\Gamma_i \models \Gamma_1 \Delta \Gamma_2$ and $\Gamma_1 \nabla \Gamma_2 \models \Gamma_i$, for $i=1,2$.

Consider again a mediated environment with a mediated schema $M=(\mathcal{A}, \Sigma_Q)$, a mediated mapping μ and, for each $k=1, \dots, n$, an export schema E_k , an import schema I_k and a local mapping γ_k .

To determine the query constraints Σ_Q , we perform two steps:

(Derivation of the Import Schema Constraints) For each $k=1, \dots, n$, obtain a set of constraints Ψ_k for the import schema I_k in such a way that γ_k induces a mapping from consistent states of E_k into consistent states of I_k .

(Computation of the Query Constraints) Compute the query constraints Σ_Q so that Σ_Q is tautologically equivalent to the g.l.b. of Φ_1, \dots, Φ_n , where Φ_k is Ψ_k after translation to the vocabulary of the mediated schema, for each $k=1, \dots, n$.

The following example illustrates the first step.

Example 3: Consider the `sales` mediated schema introduced in Figure 2. As shown in Figure 2(a), the vocabulary of `sales` consists of three classes, `s:Product`, `s:Book` and `s:Music`, and two properties, `s:title` and `s:pub`. Furthermore, Figures 2(c)-(e) indicate that the import schemas for the Amazon, the eBay and the BN export schemas have the same classes and properties as `sales`, but prefixed with “ai:”, “ei:” and “bi:”, respectively.

The derivation of the constraints of an import schema intuitively translates the constraints of the corresponding external schema to the vocabulary of the import schema, and depends on the local mappings. For example, from Fig. 2(c) and Fig. 1(b), we have that

$$\text{ai:pub} \equiv \text{a:pub} \circ \text{a:name} \quad (1)$$

$$\text{ai:Book} \equiv \text{a:Book} \quad (2)$$

$$\text{a:Book} \sqsubseteq (\geq 2 \text{ a:pub}) \quad (3)$$

$$\text{a:Publ} \sqsubseteq (\geq 3 \text{ a:name}) \quad (4)$$

$$\exists \text{ a:pub}^- \sqsubseteq \text{a:Publ} \quad (5)$$

from which we obtain the following constraint of the Amazon import schema

$$\text{ai:Book} \sqsubseteq (\geq 6 \text{ ai:pub}) \quad (6)$$

Indeed, given any interpretation \mathfrak{s} that satisfies (1) to (5):

- $\mathfrak{s}(\text{a:pub})$ associates each individual in $\mathfrak{s}(\text{a:Book})$ with at least 2 individuals in $\mathfrak{s}(\text{a:Publ})$, by (3) and (5)
- $\mathfrak{s}(\text{a:name})$ associates each individual in $\mathfrak{s}(\text{a:Publ})$ with at least 3 individuals, by (4)
- hence, $\mathfrak{s}(\text{ai:pub})$ associates each individual in $\mathfrak{s}(\text{a:Book})$ with at least 6 individuals, by (1), which is expressed in (6)

As a second example, we derive a constraint of the eBay import schema. From Fig. 2(d) and 1(d), we have that

$$\text{ei:Product} \equiv \text{e:Product} \quad (7)$$

$$\text{ei:Music} \equiv \text{e:Product} \sqcap \exists \text{ e:type.}\{\text{'music'}\} \quad (8)$$

$$\text{ei:Book} \equiv \text{e:Product} \sqcap \exists \text{ e:type.}\{\text{'book'}\} \quad (9)$$

$$\text{e:Product} \sqsubseteq (= 1 \text{ e:type}) \quad (10)$$

From (8) and (9), we have

$$\text{ei:Book} \sqsubseteq \text{ei:Product} \quad (11)$$

$$\text{ei:Music} \sqsubseteq \text{ei:Product} \quad (12)$$

Furthermore, since by assumption different constants denote different individuals (see Section 2.1), from (8), (9) and (10), we also have that

$$\text{ei:Book} \mid \text{ei:Music} \quad (13)$$

The rest of the constraints in Figs. 3(a), (b) and (c) follows likewise.

4. Modeling the query constraints

We discuss in this section how to model query constraints.

Example 4: Referring to Example 3, consider the following sets of constraints:

- Ψ_A , Ψ_E and Ψ_B – the sets of constraints of the Amazon, eBay and BN import schemas, shown in Figures 3(a), (b) and (c).
- Φ_A , Φ_E and Φ_B – the sets of constraints obtained by translating, respectively, Ψ_A , Ψ_E and Ψ_B to the vocabulary of the mediated schema. The translation is simply a process that replaces `ai:Product` by `s:Product`, and so on.

We first observe that it does not make sense to compute the g.l.b. of Ψ_A , Ψ_E and Ψ_B , since these constraints are written in different vocabularies. Therefore, we compute the g.l.b. of Φ_A , Φ_E and Φ_B , which are constraints in the same vocabulary (that of the mediated schema). Since

$$(\Phi_A \Delta \Phi_E) \Delta \Phi_B = \Phi_A \Delta (\Phi_E \Delta \Phi_B) = Th(\Phi_A) \cap Th(\Phi_E) \cap Th(\Phi_B)$$

we have to find the constraints that are simultaneously derivable from each of Φ_A , Φ_E and Φ_B .

We first analyze in detail what `minCardinality` constraints for property `s:pub` are in $\Phi_A \Delta \Phi_E \Delta \Phi_B$.

From Figures 3(a), (b) and (c), we have the following minCardinality constraints in Ψ_A , Ψ_E and Ψ_B :

$$ai:Book \sqsubseteq (\geq 6 ai:pub) \quad (\text{in } \Psi_A) \quad (14)$$

$$ei:Product \sqsubseteq (\geq 1 ei:pub) \quad (\text{in } \Psi_E) \quad (15)$$

$$bi:Book \sqsubseteq (\geq 2 bi:pub) \quad (\text{in } \Psi_B) \quad (16)$$

We also have the following subset constraint in Ψ_E :

$$ei:Book \sqsubseteq ei:Product \quad (\text{in } \Psi_E) \quad (17)$$

When translated to the vocabulary of the mediated schema, identified by the prefix “s:”, the constraints in (14) to (17) become:

$$s:Book \sqsubseteq (\geq 6 s:pub) \quad (\text{in } \Phi_A) \quad (18)$$

$$s:Product \sqsubseteq (\geq 1 s:pub) \quad (\text{in } \Phi_E) \quad (19)$$

$$s:Book \sqsubseteq (\geq 2 s:pub) \quad (\text{in } \Phi_B) \quad (20)$$

$$s:Book \sqsubseteq s:Product \quad (\text{in } \Phi_E) \quad (21)$$

Hence, the only minCardinality constraint for property $s:pub$ that is simultaneously derivable from Φ_A , Φ_E and Φ_B is

$$s:Book \sqsubseteq (\geq 1 s:pub) \quad (\text{in } \Phi_A \Delta \Phi_E \Delta \Phi_B) \quad (22)$$

Indeed, we have that:

- (18) implies (22) and (20) implies (22), if we observe that a minCardinality of m implies a minCardinality of n , if $m \geq n$
- (19) and (21) imply (22)

In fact, the constraint in (22) is the only cardinality constraint in $\Phi_A \Delta \Phi_E \Delta \Phi_B$, since the constraint in (16) is the only cardinality constraint in Ψ_B (see the middle column of Figure 3(c)).

The subset constraints in $\Phi_A \Delta \Phi_E \Delta \Phi_B$ are those shown in the third column of Figure 3(d); in fact, they are in the intersection of Φ_A , Φ_E and Φ_B . Furthermore, since Ψ_B and, consequently, Φ_B have no exclusion constraint, so does $\Phi_A \Delta \Phi_E \Delta \Phi_B$.

The domain and range constraints in $\Phi_A \Delta \Phi_E \Delta \Phi_B$ are those shown in the first column of Figure 3(d); in fact, they are in the intersection of Φ_A , Φ_E and Φ_B , except for the domain constraint $\exists s:pub \sqsubseteq s:Product$, which is derived as follows.

From Figures 3(a), (b) and (c), we have the following domain constraints in Ψ_A , Ψ_E and Ψ_B :

$$\exists ai:pub \sqsubseteq ai:Book \quad (\text{in } \Psi_A) \quad (23)$$

$$\exists ei:pub \sqsubseteq ei:Product \quad (\text{in } \Psi_E) \quad (24)$$

$$\exists bi:pub \sqsubseteq bi:Book \quad (\text{in } \Psi_B) \quad (25)$$

We also have the following subset constraints in Ψ_A and Ψ_B :

$$ai:Book \sqsubseteq ai:Product \quad (\text{in } \Psi_A) \quad (26)$$

$$bi:Book \sqsubseteq bi:Product \quad (\text{in } \Psi_B) \quad (27)$$

When translated to the vocabulary of the mediated schema, identified by the prefix “s:”, the constraints in (23) to (27) become:

$$\exists s:pub \sqsubseteq s:Book \quad (\text{in } \Phi_A) \quad (28)$$

$$\exists s:pub \sqsubseteq s:Product \quad (\text{in } \Phi_E) \quad (29)$$

$$\exists s:pub \sqsubseteq s:Book \quad (\text{in } \Phi_B) \quad (30)$$

$$s:Book \sqsubseteq s:Product \quad (\text{in } \Phi_A) \quad (31)$$

$$s:Book \sqsubseteq s:Product \quad (\text{in } \Phi_B) \quad (32)$$

Hence, the domain constraint for property $s:pub$ that is simultaneously derivable from Φ_A , Φ_E and Φ_B is

$$\exists s:pub \sqsubseteq s:Product \quad (\text{in } \Phi_A \Delta \Phi_E \Delta \Phi_B) \quad (33)$$

Figure 3(d) shows the final set of query constraints. Note that we indeed have that

$$\Phi_X \models \Phi_A \Delta \Phi_E \Delta \Phi_B \quad \text{for } X \in \{A, E, B\} \quad (34)$$

For example, after translation to the vocabulary of the mediated schema, the constraints of the Amazon import schema, shown in Figure 3(a), indeed imply the query constraints of the mediated schema, shown in Figure 3(d). Thus, the data returned from the Amazon data source will always satisfy the query constraints of the mediated schema, and likewise for the other data sources.

This illustrates the computation of the query constraints of a mediated schema as the g.l.b. of the sets of constraints of the import schemas, after proper translation.

5. Conclusions

We argued in Section 4 that, as far as queries are concerned, the constraints of the mediated schema should be modeled as the least upper bound of the sets of constraints of the export schemas, after appropriate translation to a common vocabulary. This assures that applications that use the mediated schema will correctly interpret query results.

Finally, we are in the process of incorporating the results reported in this paper into a mediator framework, as part of a larger effort to create mediators for data sources available on the Web [10].

6. Acknowledgments

This work was partly supported by CNPq under grants 140417/05-2, 301497/06-0, 473110/08-3 and 620143/2008-8.

7. References

- [1] Atzeni, P.; Cappellari, P.; Torlone, R.; Bernstein, P.A.; Gianforme, G. (2008) “Model-independent schema translation”. The VLDB Journal. v.17, n.6, pp. 1347-1370.
- [2] Baader, F.; Nutt, W. (2003) “Basic Description Logics”. In: Baader, F.; Calvanese, D.; McGuinness, D.L.; Nardi, D.; Patel-Schneider, P.F. (Eds) The Description Logic Handbook: Theory, Implementation and Applications. Cambridge University Press, Cambridge, UK.

- [3] Breitman, K., Casanova, M., and Truszkowski, W. (2007) *Semantic web: concepts, technologies, and applications*. Springer, London.
- [4] Calvanese, D.; De Giacomo, G.; Lembo, D.; Lenzerini, M.; Poggi, A.; Rosati, R.; Ruzzi, M. (2008). "Data Integration through DL-Lite-A Ontologies". In: Proc. Third International Workshop on Semantics in Data and Knowledge Bases (SDKB 2008), pp. 26-47.
- [5] Cali, A.; Calvanese, D.; De Giacomo, G.; Lenzerini, M. (2004) "Data integration under integrity constraints", *Information Systems*, Vol. 29, Issue 2, pp. 147-163.
- [6] Chiticariu, L., Kolaitis, P. G., and Popa, L. (2008) "Interactive generation of integrated schemas". In: *Proc. 2008 ACM SIGMOD Int'l. Conf. on Management of Data* (Vancouver, Canada, June 09 - 12, 2008). SIGMOD '08. ACM, New York, NY, pp. 833-846.
- [7] Curino, C.A.; Moon, H.J.; Zaniolo, C. (2008) "Graceful database schema evolution: the PRISM workbench". In: Proc. of the VLDB Endowment, v.1, n.1, pp. 761-772.
- [8] Euzenat, J. and Shvaiko, P. (2007) *Ontology matching*. Springer-Verlag.
- [9] Fagin, R.; Kolaitis, P. G.; Popa, L.; Tan, W.-C. "Quasi-inverses of schema mappings". In: Proc. PODS '07, pp. 123-132.
- [10] Lauschener, T., Casanova, M.A., Vidal, V.M.P., Macedo, J.A. "Efficient Decision Procedures for Query Containment and Related Problems". In: XXIV Simpósio Brasileiro de Banco de Dados, 2009, Fortaleza. Anais do XXIV Simpósio Brasileiro de Banco de Dados. Porto Alegre: Sociedade Brasileira de Computação, 2009.
- [11] Leme, L.A.P.; Casanova, M.A.; Breitman, K.K; Furtado, A.L. (2009) "Instance-based OWL Schema Matching". In: Proc. 11th Int'l. Conf. on Enterprise Information Systems, Milan, Italy.