



# Are Books Events? Ontological Inclusions as Coercive Sub-Typing, Lexical Transfers as Entailment

Bruno Mery, Christian Retoré

## ► To cite this version:

Bruno Mery, Christian Retoré. Are Books Events? Ontological Inclusions as Coercive Sub-Typing, Lexical Transfers as Entailment. LENLS: Logic and Engineering of Natural Language Semantics, Nov 2015, Tokyo, Japan. pp.74-87. lirmm-01311120

**HAL Id: lirmm-01311120**

**<https://hal-lirmm.ccsd.cnrs.fr/lirmm-01311120>**

Submitted on 3 May 2016

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

# Are Books Events?

## Ontological Inclusions as Coercive Sub-Typing, Lexical Transfers as Entailment

Bruno Mery and Christian Retoré

LaBRI & Université de Bordeaux,  
LIRMM & Université de Montpellier

**Abstract.** <sup>1</sup> We present a way to combine coercive sub-typing relations, used to model ontological inclusions, and entailment using polymorphic terms provided by the lexicon in order to provide a coherent calculus integrated in  $\Lambda TY_n$ . The resulting system is able to parse different types of lexical phenomena. We also illustrate differences of assumptions between related theories of lexical meaning, using the different ways to define coercive sub-typing and type systems as examples.

**Keywords:** Lexical Semantics, Type Theory, Coercive Sub-Typing, Composition, Ontological Inclusion.

### Introduction

Human language is highly polysemic, and present many challenges to the compositional semantics in the tradition of [Montague, 1974]: ambiguous words, vagueness, context-dependent usages, creative use of words, hypostasis. . . In publications such as [Moravcsik, 1982] and [Cruse, 1986], many of these issues have been spelt out, and the Generative Lexicon Theory have been proposed as a guideline of integration of lexical data into compositional semantics in [Pustejovsky, 1995]. More recently, several systems using different type theories for the purpose of modelling lexical semantics have been proposed. Those include, among others: Asher’s Type Composition Logic presented in [Asher, 2011], Bekki’s Dependent Type Semantics presented in [Bekki, 2014], Cooper’s Type Theory with Records presented in [Cooper, 2007], Luo’s Unified Type Theory presented in [Luo, 2011], as well our framework  $\Lambda TY_n$ , developed in [Mery and Retoré, 2013, Retoré, 2014].

---

<sup>1</sup> Many thanks to the attendants of the TYTTLES workshop (ESSLLI Barcelona, August 2015), and to many people for long and helpful discussions on the topic of type-theoretic lexical semantics, including but not limited to Asher, Bekki, Chatzikyriakidis, Cooper, Icard, Kubota, Levine, Luo, Mineshima, Moot, Pustejovsky, Real, Satoh, Tanaka, . . .

Special thanks to Frédéric Lang, who did a lot of preliminary work on the subject, and to Sergei Soloviev, for clarifications on the history of sub-typing formalisms.

Over the last several years, the specific issue of sub-typing has been examined in several of these settings, as in [Bekki and Asher, 2012] and [Luo et al., 2013], for instance.

In the present paper, we present a way to combine simple mechanisms, using sub-typing to represent *ontological inclusions only*, while other lexical phenomena are modelled using optional terms derived from the lexicon and entailment. We present rules (originally provided for the simply-typed  $\lambda$ -calculus) for establishing coercive sub-typing relations in  $\Lambda TY_n$ , a system able to derive formulae of predicate calculus with the desired formal results. We precise the combination of sub-typing and standard entailment, forming a coherent calculus able to articulate the composition of meaning from syntax to semantics and logical representation. Thereafter, we survey the other common uses of “coercive sub-typing” in different formal frameworks for lexical semantics, clarifying what is meant by *coercion*. This allows us to illustrate some of the differences between two classes of type-theoretic systems for lexical semantics. While we do not claim that the choices presented here are the only possible ones, we want to argue that they are adequate for our purposes.

## 1 Coercive Sub-Typing and Entailment in Lambda TYn

### 1.1 Formal and Applicative Context

The goal of our  $\Lambda TY_n$  is to integrate lexical phenomena such as restriction of selection and polysemy, presented in theories such as the Generative Lexicon (see [Pustejovsky, 1995]), into Montague-style compositional analysis. As in that tradition, we analyse utterances made in natural language starting from the syntax and semantics to produce a logical representation in the form of a formula of (higher-order) predicate calculus, suitable for interpretation in game semantics, model-theoretic semantics and many other theories. The integration of lexical phenomena is largely made by the distinction of many *sorts* or base types, in contrast with the standard single-sort  $\mathbf{e}$  in Montague; having a collection of sorts  $\mathbf{e}_1, \mathbf{e}_2 \dots \mathbf{e}_n$  is a way to distinguish *selection classes* as might appear in a dictionary. We have noted that, in *Le Petit Robert* (in French) and the online *wiktionary* (in English), the entry for the verb *to bark* is noted to be *said of animals, especially dogs*<sup>2</sup>. We need to be able to distinguish such selection classes in order to decide if a restriction of selection is respected (as in *the dog barked*), violated (as in *\*the chair barked*), or indicates that some lexical phenomena is in play (as in *the sergeant barked*, which indicates a certain judgement from the speaker).

In our framework, and in others for lexical semantics that rely on a many-sorted type theory, *sub-typing* is of particular interest. Relations between types can allow the system to accommodate some properties of selection classes. Specifically, we argue that it is quite suited to the represent ontological inclusions (“is-a” relations, named “hyponymy” in the linguistic literature when denoting relations between *words*).

<sup>2</sup> Many other dictionaries note the differences between conventional meaning depending on whether the subject is a dog, an animal other than a dog, or an human being. More details on selection classes, dictionaries and lexical semantics are given in section 2.3.

For instance, if we define  $Apple < Fruit$  and  $Fruit < Food$  as types with a simple hierarchy, if the predicate *to eat* is defined to apply to patients of type *Food*, it will apply to objects that are of type *Apple* as well.

The foundations of sub-typing as a tool for lexical semantics have been under discussion since [Pustejovsky, 1995] and coercive sub-typing rules have been proposed for Martin-Löf Type Theories as recently as [Luo et al., 2013]. For  $\Lambda TY_n$ , we rely on System-F, the  $\lambda$ -calculus with polymorphic types. Attempts have been made since the late 80's to equip System-F with a sub-typing relation, in order to present a formal system of computation equivalent to most current programming languages, culminating in [Cardelli et al., 1991]; such systems remain complex (and not really satisfactory) in nature. Subsequent efforts have relied on extending System-F and detailing how a sub-typing relation defines a logic system akin to entailment, as in [Longo et al., 2000]. In the present article, we will **not** attempt to provide sub-typing rules for higher-order types, but will argue that a simple version of the sub-typing relation (i.e., coercive sub-typing as an entailment judgement for simply-typed terms) is sufficient for our purposes.

*Coercive sub-typing* is a relatively new means to formalise sub-typing relations ([Luo et al., 2013] uses coercive sub-typing). Building a coercive sub-typing relation requires a collection of base types (the *sorts*,  $\mathbf{e}_i$ ) and of *coercions* – given as  $\lambda$ -terms – between some of those sorts, reflecting ontological inclusions, written  $c_{ij} : \mathbf{e}_i \rightarrow \mathbf{e}_j$ . The collection of coercions should form a *partial order* on sorts, such as there is at most one coercion for every (ordered) pair of sorts (if the coercion  $c_{ij}$  is defined, it is noted that  $\mathbf{e}_i < \mathbf{e}_j$ ) and that the graph of the relation forms, at most, a directed acyclic graph. Note that “diamond formations” in which are defined coercions  $c_{in}, c_{im}, c_{nj}, c_{mj}$  (i.e., it is true that  $\mathbf{e}_n < \mathbf{e}_i$ ,  $\mathbf{e}_m < \mathbf{e}_i$ ,  $\mathbf{e}_j < \mathbf{e}_n$  and  $\mathbf{e}_j < \mathbf{e}_m$  with four different types) are only allowed when  $c_{in} \circ c_{nj} = c_{im} \circ c_{mj}$ ; i.e., when all possible compositions of coercions yielding the same type are equal. In that case, there is one single well-defined composed coercion,  $c_{ij}$ , and it can be unambiguously written that  $\mathbf{e}_j < \mathbf{e}_i$  (Soloviev, pc.). It is of course possible to have a formulation that does not contain such formations; that issue is, of course, very similar to allowing or not multiple inheritance in object-oriented languages. Observe that ontological inclusions do respect that unicity requirement, as they represent “is-a” relations (if an object is an *Apple*, it is also a *Fruit* – the coercion involved is object identity).

Given suitable rules, this axiomatic partial order is extended to complex types (e.g. functional types). The result is a *coercive* sub-typing when:

1. between any two (base or complex) types  $X, Y$  such as  $X < Y$ , there is an unique coercion<sup>3</sup> that turns inhabitants of  $X$  into objects of type  $Y$ , and this coercion can be reconstructed from the coercions given between base types, and
2. the extended relation remains acyclic.

<sup>3</sup> The coercion can be described by a  $\lambda$ -term, unique w.r.t. a suitable notion of equality, such as “modulo  $\beta - \eta$  equivalence”; in practice, this term is often implicit and omitted.

## 1.2 Definition and Propagation of Sub-Typing

Coercive sub-typing can be viewed as an abbreviation: when having an object of  $x : X$  applied to  $f : Y \rightarrow U$  for some result type  $U$ , if  $X < Y$ , then there is a unique coercion  $c$  that can be used in order to have  $(f (c x)) : U$ , a well-formed term. In such cases, the (unambiguous) coercion can be omitted and the term written as  $(f x)$ .

Formally, *Coercive application* is properly defined as:

$$\frac{\vdash x : A \vdash f : B \rightarrow U \quad A < B}{\vdash f(x) : U}$$

As seen above, coercive sub-typing also requires a set of *axioms* in the form  $\mathbf{e}_i < \mathbf{e}_j$  for suitable sorts; each sub-typing axiom is an abbreviation for a coercion  $c_{ij} : \mathbf{e}_i \rightarrow \mathbf{e}_j$ , respecting the properties of coherence (at most one coercion from a sort to another – including composed coercions) and acyclicity.

The following rules are used to extend the sub-typing relation  $<$  to complex types:

**Sub-typing transitivity**  $\frac{A < B \quad B < C}{A < C}$

**Co-variance and contra-variance of implication**  $\frac{A < B}{T \rightarrow A < T \rightarrow B}$

**Simple co-variance**  $\frac{A < B}{B \rightarrow T < A \rightarrow T}$

**Simple contra-variance**  $\frac{A < B \quad C < D}{D \rightarrow A < C \rightarrow B}$

These rules are sufficient, and correspond to the rules of sub-typing for simply-typed  $\lambda$ -calculus. It was demonstrated in [Lang, 2014] that, for a larger set of rules (including the propagation of sub-typing over some form of type quantification), the  $<$  relation is *coercive*, that is, that coherence and acyclicity hold; they therefore also hold for the present set of rules. Accordingly, if there is a coercion  $c_{X \rightarrow Y}$  that can be constructed using these rules, then there is a unique canonical way to build it from axiomatic coercions  $c_{i \rightarrow j}$ , which is detailed in [Retoré, 2014].

## 1.3 Combining Sub-Typing and Entailment in a Coherent Calculus

The relation  $<$ , defined above for the simply-typed  $\lambda$ -calculus, is a special case of the usual entailment  $\vdash$ . Specifically, the “coercive application”  $\frac{\vdash x : A \vdash f : B \rightarrow U \quad A < B}{\vdash f(x) : U}$  can be simulated by the following:

$$\frac{\vdash x : A \vdash f : B \rightarrow U \quad \vdash c : A \rightarrow B}{\vdash f(c(x)) : U}$$

In other words, the relation of coercive sub-typing is a sub-calculus of System-F, the syntax of which is used for  $ATY_n$ . A deduction in our system with coercive sub-typing can thus combine  $<$  and  $\vdash$  proof-trees seamlessly in order to form the proof corresponding to the logical representation of a sentence.

## 1.4 A Purely Logical Syntax and Semantics Analysis

We will be using the example sentence *John began a book*.

This sentence, while simple in itself, illustrates the following phenomena:

- *Epsilon quantification* (with the article *a*); choice functions are also suitable candidates for this form of selection. The article *a* is polymorphic and, when applied to a sort  $X$  and a predicate  $P^{X \rightarrow \mathbf{t}}$ , yields a (new) object satisfying  $P$ ; see [Mery et al., 2015] for details.
- *Qualia adaptation* with the telic of *book*; the *telic quale* is a classic use of lexical items in the Generative Lexicon in which the most common usage for an object is substituted for it (here, the event of *reading* the book in question).
- *Ontological inclusion*, the topic of this article (reading events are a sub-type of events).

We will use coercive sub-typing solely for ontological inclusions, and entailment with terms provided by the lexicon for the other phenomena. The semantic sorts used here are:  $H$  for human beings,  $RM$  for reading materials (books, magazines...),  $Evt$  for events,  $REvt$  for reading events. The base type  $\mathbf{t}$  for propositions is also used.

**Syntax analysis:** using a Lambek-style grammar and lexicon, the sentence is parsed as in the Montagovian tradition:

$$\frac{\text{John} : NP \quad \frac{\text{began} : (NP/S) \backslash NP \quad \frac{a : NP/N \quad \text{book} : N}{a \text{ book} : NP}}{\text{began a book} : NP/S}}{\text{John began a book} : S}$$

**Syntactic to semantic terms:** the syntactic terms (categories of the grammar) are translated systematically into semantic terms ( $\lambda$ -terms) using a complex lexicon. As in most accounts of lexical semantics with many sorts, type mismatch indicates a violation of restriction class. We provide two mechanisms for the resolution of such mismatches: using coercive sub-typing, or using explicit terms introduced by the lexicon. Accordingly, additional terms might be provided (in this case, associated to the lexical entry for *book*, the term  $f_{Telic} : RM \rightarrow REvt$  is provided as a lexical transformation giving access to the event of reading a book) in order to account for semantic phenomena in a co-compositional way; the general hierarchy of types also provides relevant sub-typing judgements (here,  $REvt < Evt$ ). Like in standard Montague semantics, the syntactic proof is transformed into a semantic proof. However, complying with the additional information from the lexicon, the resulting proof might be more complicated and is not always valid; we do not want to give an interpretation to infelicitous sentences. The syntax-semantics correspondence is as follows:

Lexemes, Syntax	Semantics	Remarks
John, $NP$	$j : H$	
began, $(NP/S) \backslash NP$	$be : IIX.$ $X \rightarrow (X \rightarrow Evt) \rightarrow H \rightarrow \mathbf{t}$	The first syntactic argument can be anything, given a suitable lexical transformation to an event.
a, $NP/N$	$\varepsilon : IIX.(X \rightarrow \mathbf{t}) \rightarrow X$	$A$ is a polymorphic Hilbert operator as in [Mery et al., 2015].
book, $N$	$b : RM \rightarrow \mathbf{t}$	Common nouns are predicates.

**Meaning assembly:** the standard process of natural deduction combines proofs including  $\vdash$  and, if needed,  $<$ , in order to derive a term of type  $\mathbf{t}$  that corresponds to the sentence. In the example above, this will yield the following semantic proof:

$A$  : (Specialisation of *began* for the type *Reading Material*.)

$$\frac{\vdash \text{be} : \Pi X. X \rightarrow (X \rightarrow \text{Evt}) \rightarrow H \rightarrow \mathbf{t}}{\vdash (\text{be } \{RM\}) : RM \rightarrow (RM \rightarrow \text{Evt}) \rightarrow H \rightarrow \mathbf{t}}$$

$B$  : (Specialisation of the polymorphic  $\varepsilon$  for the same type.)

$$\frac{\vdash \varepsilon : \Pi X. (X \rightarrow \mathbf{t}) \rightarrow X}{\vdash (\varepsilon \{RM\}) : (RM \rightarrow \mathbf{t}) \rightarrow RM}$$

$C$  : (Derivation of *began a book* without qualia or sub-typing.)

$$\frac{A \quad B \quad \vdash b : RM \rightarrow \mathbf{t}}{\vdash (\text{be } \{RM\}) (\varepsilon \{RM\} b) : (RM \rightarrow \text{Evt}) \rightarrow H \rightarrow \mathbf{t}}$$

$D$  : (The *telic* quale:)  $\vdash f_{\text{Telic}} : RM \rightarrow \text{REvt}$

$E$  : (The sub-typing derivation.)

$$\frac{\text{REvt} < \text{Evt}}{RM \rightarrow \text{REvt} < RM \rightarrow \text{Evt}}$$

$$\frac{\frac{C \quad D \quad E}{\vdash (\text{be } \{RM\}) (\varepsilon \{RM\} b) f_{\text{Telic}} : H \rightarrow \mathbf{t}} \quad \vdash j : H}{\text{(The full sentence:)} \quad \vdash (\text{be } \{RM\}) (\varepsilon \{RM\} b) f_{\text{Telic}} j : \mathbf{t}}$$

**In summary**, the resulting calculus is fully integrated in  $\Lambda TY_n$ . Being a sub-calculus of System-F, it is well-founded and strongly normalising; thus, if the final semantic term is of type  $\mathbf{t}$ , it also constitutes a valid formula of predicate calculus. Everything is compliant with our guiding principle: to be able to establish the semantic representation of utterances from their syntax giving a formula of predicate calculus as a result, detailing every step of the way. At the end, the result is, as usual, a formula of (potentially higher order) predicate calculus that can be interpreted easily in any suitable theory, including model-theoretic semantics.

## 2 Differences between Type-Theories of Lexical Semantics

We want to illustrate some differences between two classes of systems: our proposal,  $\Lambda TY_n$ , and MTT-based frameworks for type-theoretic lexical semantics. There are, of course, many similarities between our system and the others: all aspire to be able to model the lexical phenomena detailed in [Pustejovsky, 1995] via type-driven mechanisms.

The goal of all frameworks representing lexical semantics in type theory is to be able to derive a representation of the meaning(s) of the felicitous utterances, and to the correct rejection of the infelicitous ones. This is why every system makes some assumptions as to the *selection classes* of lexical items, that are absent from classical compositional semantics (in [Montague, 1974], every syntactically correct utterances has a semantic representation). There are many ways to characterise the selection classes; as we will discuss in Section 2.3, dictionaries are can provide some insight on this point. But listing restrictions of selection can only be the start, as they are routinely softened, adapted, accommodated using lexical mechanisms (predicates that are listed as applying to *Human* beings, such as *to be joyful*, can also be applied to cities); those *meaning shifts* are what concern all frameworks of lexical semantics, and are modelled via *type operations* in all of these systems (some semantic terms will have their type changed, coerced, or made compatible with others). *Co-predications* have proven a difficult issue, as it is both difficult to have terms being of different types at once (all of the systems involved have specific solutions), and to get information on whether a co-predication is valid or not; such informations are not available in dictionaries.

Our system differs from MTT-based frameworks in several ways. There are several systems that are inspired by Martin-Löf’s Type Theory and derivatives (sometimes named “Modern” Type Theories, or Type Theories with Canonical Objects). They include Asher’s TCL presented in [Asher, 2011] (though TCL presents many aspects that are closer to Montague than many other formalisms), Bekki’s DTS presented in [Bekki, 2014], Cooper’s TTR presented in [Cooper, 2007], and Luo’s UTT presented in [Luo, 2011]. They incorporate complex typing information, at least on the form of Dependent Types, and common nouns are typically considered to be atomic types, with a high or even unbounded number of lexical sorts. Each lexical entry is given as a single type and (sometimes) term. Composition rules are quite complex, but effective; MTT-based systems have proven very apt at tasks such as reasoning, predicting or rejecting inferences, and learning, and there are several working implementations. However, there are no direct link to an analysis of syntax, and syntactic information is vital for establishing the boundaries of local phenomena.

$\Lambda TY_n$  has a different approach as to the organisation of the lexicon. Each lexical entry is a main  $\lambda$ -term with a type, associated with some (or none) optional terms that provide all lexical information. We have a limited (and finite) number of sorts, and the rules of composition are abstraction and application (of terms and types). Note that we have described  $\Lambda TY_n$  using the syntax of System-F, but that is not a formal requirement; we would be able to give the same system using MTT, DTS, UTT, or even the simply-typed  $\lambda$ -calculus (the enumeration of optional terms would be burdensome in the latter case, but manageable, as every use of polymorphic terms is made over a finite family of types). Our main goal was to keep close to Montague, as by analysing the syntax of the utterances using categorial grammars, we are able to derive semantic representations seamlessly. We do not claim that this is the only way, or even that we are able to represent all possible phenomena in our systems, but we think that the approach is sound.

In the following, we discuss the specific differences as to the use of coercive subtyping in MTT-based systems and as in presented in this article.



## 2.1 Different “Coercions” in Lexical Semantics

On the one hand, the examples used above illustrate an intuitive notion to link to sub-typing, namely *ontological inclusion*. This is the simplest linguistic phenomenon covered by coercive sub-typing: the coercion is that of “asymmetrical identity” or “is-a” relation, familiar to any database modeler, forming a *directed acyclic graph* (more precisely a rooted tree: the taxonomy). Coherence and acyclicity are thus guaranteed; when there is a coercion between any two types, it is *unique* (modulo equivalence) and *canonically defined* from axiomatic coercions.

On the other hand, the word “coercion”, applied to lexical semantics, is often used in a much broader sense: it is the use of a word that is associated with a primary type in the lexicon, that the context of enunciation “coerces” into another type. That notion of “linguistic coercion” subsumes ontological inclusions (“accommodation” according to Pustejovsky), but also qualia adaptation, grinding, and many other lexical phenomena.

An extreme example from [Nunberg, 1993] is *the ham sandwich left without paying*, a meaning shift that coerces *Food* items into the *Human* beings that have ordered them in conversations among restaurant staff. We differentiate the two notions and argue that using coercive sub-typing for ontological inclusions, but not for other forms of linguistic “coercions”, allows us to have a complete model of lexical phenomena that is coherent with our assumptions. Other systems for lexical semantics do differently.

## 2.2 Martin-Löf-inspired Systems

In MTT-based systems, a lot of the lexical information is encoded in the types themselves, with complex relations indicated in the dependencies between terms and types. In some cases, the interpretation of such terms is much less straightforward than with predicate calculus. Systems such as TTR that encode all information as typing judgements, for example, have issues with logical connectives such as negation (negation in predicate calculus is simply a predicate over propositions, while negating a type will result in a type that has no inhabitants in the usual interpretation); these issues sometimes require a very different interpretation than usual. In most of those systems, coercive sub-typing becomes a vital component, as it could be used to reflect all the (general) linguistic coercions.

Of course, doing this without would result in very a disruptive notion of sub-typing; none of the systems above do this. Some issues are as follow: recall that a well-behaved (normal and complete) system of coercive sub-typing is such that, between two given types, if one is a sub-type of the other, there is a single canonical coercion between the two, and that the graph made by elementary coercions does not contains cycles. With general linguistic coercions, there not only several ways to coerce one type into another, but it is highly desirable to have those different coercions co-exist in the type system.

Between *Books* and *Events* (the example above, which is a favorite of Asher), there are more than one lexical coercion: at least two that refer to a *reading* and a *writing Event*, but also in suitable contexts *printing*, *colouring*... Distinguishing between the different lexical coercions is a necessary feature of any system for lexical semantics.

If such relations can be defined globally over types, it is not clear what the objects inhabiting those types are: when there are distinct ways to convert a *Book* into an *Event*, then the individual of type *Book* does not have a formal definition when used in a context requiring an *Event*!

Moreover, it is easy to find a sequence of non-trivial coercions making a cycle. *The ham sandwich left without paying* is evidence of a coercion from *Food* to *Human*. There is an inclusion of *Human* in *Animal*, and *the salmon we had yesterday was delicious* is evidence of a grinding coercion from *Animal* to *Food*. If all coercions here were sub-typing coercions, we would have constructed a non-trivial morphism from *Food* to *Food*.

Most proposals that have been made, proposing that (general) linguistic coercions can be modelled with sub-typing relations, have ways to deal with these problems. [Bekki and Asher, 2012] proposes a system for dealing with co-predications using injections that coerce the types of functors, rather of their argument. That strategy can of course work, but the authors of [Bekki and Asher, 2012] only hints at a formal system, that we think can be given within  $\Lambda TY_n$ . Moreover, they propose a much more general system than is usual for sub-typing, using only functor modifiers and the polymorphic application rule, in which it would be hard to demonstrate the coherence and acyclicity of the relation.

In [Asher and Luo, 2012], on the other hand, that very solution seems to be rejected, and a very useful notion is proposed: *local coercions*. Here, the coercive judgements of the form  $A < B$  are not general axioms, but specified in the context of a specific derivation (as in  $A < B \vdash f : A \rightarrow B \ a : A \dots$ ). Thus, at any given point, the sub-typing system can maintain its coherence and acyclicity. This means that the ontology does not specify a system of global relations between types, but that those are local to specific utterances and context-dependant when parsing utterances. [Asher and Luo, 2012] extends this proposal further by having coercions written as terms with dependent types. A full, and rather complex, formalisation is given in [Luo et al., 2013] for general type theories, with no specific application to linguistics.

## 2.3 Our Montagovian System

The original goal of Montagovian Generative Lexical Theories such as  $\Lambda TY_n$  and, to some extent, Asher's TCL, is to model lexical phenomena starting from *restriction of selection* added to the standard logical Montague analysis of syntax and semantics. Grammatical sentences such as *the chair barked* are infelicitous because the predicate *bark* apply to *animals*, and *chairs* are not *animals*. Restriction of selection needs a system of types that can distinguish between furniture and animals – the base types are selection classes, named *sorts*, and the types can also define relations of ontological inclusions. We believe the sorts involved to number a few hundred at most; this is not an absolute necessity and should be refined by linguistic studies. This intuition is based on several points of (admittedly anecdotal) linguistic evidence.

On the one hand, *classifiers*, in language that have them, form an interesting hierarchy of categories which, while not complete (as classifiers abound for physical objects and agents, but are sparse concerning abstract entities), is very broad, multi-scalar (with a few generic classifiers, a number of “intermediate” classifiers and many highly specialised ones), and provide linguistic distinctions between classes of objects that are remarkably similar across languages (see [Mery and Retoré, 2013] and upcoming publications for details).

On the other hand, the readily available linguistic references (common dictionaries) provide *some* insight as for the possible selection classes for predicates.

In the early studies of lexical semantics, [Moravcsik, 1982], [Cruse, 1986] and [Pustejovsky, 1995], the traditional dictionaries have been characterised as completely inadequate to represent language. To paraphrase Pustejovsky, as dictionaries are *sense-enumeration* lexica, they can, at best, represent a fragment of language at a specific point of time.

Dictionaries are unable to account for the ability to give new meaning to words by combining them in novel, contextually-appropriate ways. We subscribe to this view, but argue that a perfect dictionary would give us the common restriction classes for predicates and thus a basic system of sorts, while our (generative) lexicon will provide the means to explore if a violation of selection is motivated by a lexical mechanism, or simply invalid. No dictionary would list all *words* that can apply to a specific predicate, and these restrictions of selection, when they are present, are by no means complete or easy to extract computationally. Nevertheless, *Larousse* in French commonly uses broad selection classes including *person, man, woman, animal, (physical) body*, etc. *Littré* refines categories more finely, with predicates applying to specific materials (*glass, wool* are base categories), places, instruments, etc.

Having different base types, we wish to be able to distinguish different cases of ill-formed utterances:

- A-grammatical utterances, when rules of syntax are not respected.
- A-semantic utterances, when the syntactic terms cannot be transformed into semantic terms. (This should not happen for sentences that are grammatical if the lexicon is complete.)
- Infelicitous utterances, when there is an irresolvable type mismatch between terms (such as *\*The chair barked*).
- Utterances without interpretation, such as *The king of France is bald*.

In  $ATY_n$ , lexical phenomena other than ontological inclusions, such as qualia adaptations or facets, are not defined by the type system, but rather licensed by the lexicon; thus the explicit addition of transformative terms in the definition of each lexeme, to be used at the stage of meaning composition by the means of simple application (of terms or types) and suitably polymorphic types. We will then be able to distinguish between semantic, but infelicitous sentences (when there is no possible way to resolve a type mismatch) and felicitous sentences (when there is a semantic term available to resolve every type mismatch). Note that infelicitous sentences might be felicitous in another context, where adaptative terms are provided.

We restrict the use of coercive sub-typing to ontological inclusions, as the two concepts can be modelled using similar constraints (most importantly unicity, which is guaranteed with identical objects); as illustrated above, coercive sub-typing, using only rules corresponding to the simply-typed  $\lambda$ -calculus, is a special case of entailment, and the full calculus is thus well-defined and strongly normalising.

As we have seen, that system is designed to be practical, with an analysis completely defined from syntax to semantics. When the meaning assembly is finished, if the resulting  $\lambda$ -term is of type  $\mathbf{t}$ , it is a meaningful formula of the predicate calculus that can be used model-theoretically.

Disregarding the final typing, all lexical information, as well as quantification, selection, negation, and modalities, are presented in the form of predicates that can be evaluated. The process is also straightforward (if not trivial) to implement in functioning parsers for natural language.

## 2.4 Idiosyncratic Divergence

While we are confident that all the systems presented here can satisfactorily deal with similar issues of lexical semantics, the differences of formulations and interpretations suggest deeper divergences.

One obvious reason for the differences is with the background of their respective authors and the nature of the results. Systems inspired by Martin-Löf's Type Theory have successfully been implemented in proof-checkers and theorem-provers such as Coq and Plastic, or using pure functional languages such as Haskell, and are actively used to compute logical inferences adapted from natural language. However, these systems have no means to generate the semantic representation of utterances directly from syntax. On the other hand,  $\Lambda TY_n$  is suitable to produce logical forms from syntax, but has not been used for tasks such as learning or inference. Several tests for complex phenomena (including co-predications, qualia adaptation, virtual traveler, plural and mass readings) have been made using syntax-semantics parsers such as Grail, a wide-coverage, CG-based parser (using  $\lambda$ -DRT rather than System-F) from Richard Moot, presented in [Moot, 2010].

Another reason is, perhaps, a difference of approach to the linguistic phenomena involved. We view general linguistic coercions as *idiosyncratic* by nature, with different coercions occurring in different languages (it has been proposed that similar words in different languages do sometimes not license the same lexical phenomena), communities (the same words in the same language acquire different meanings in professional contexts, such as *operation* for scientists, medical workers and military officers) and literary genres (in Fable, Fantasy and Mythology, the type *Person* is usually much broader than *Human*). We reflected this by associating the transformations with the *terms* given in the lexicon and not their *types*: types serve to guide such systems, not specify every possible utterance. Our choice of having “a few” different sorts to act as selection classes imposes us to rely almost exclusively on lexical information, with terms having very different behaviours typed similarly. On the other hand, Martin-Löf-inspired systems rely heavily on typing to provide or block coercions; some systems provide information from the lexicon or different mechanisms, others encode everything in typing judgements or relations.

Those systems are of course able to give an account of idiosyncrasy as well, but we do not think that the implied typing mechanisms are similar: sorts are associated with words (with every common noun, or even more), which is subtly different from linguistic concepts – certainly at the granularity level, but also when defining selection classes and other lexical phenomena.

The most striking difference is thus the different options being pursued for the granularity of the basic sorts. This debate is not new, and certainly not settled; see [Goddard and Wierzbicka, 2007] (among many other references). Common answers include:

- The single sort *e*, from [Montague, 1974]. This is not sufficient to provide mechanisms for the treatment of lexical phenomena; in fact, it is not sufficient to *detect* lexical phenomena at the phase of meaning assembly.
- Some basic ontological categories (two rough dozens) distinguishing between physical/abstract, animated/inanimate, human/animal... that are most commonly distinguished in sentences.
- Classes corresponding to different linguistic concepts (a few hundreds); we have suggested classifiers as a basis in [Mery and Retoré, 2013].
- Every common noun, as proposed in [Luo, 2012].
- Every constructible type as a “basic” sort.

In some of the proposals, the base types are further organised using a multi-scalar hierarchy, with type universes, kinds and sub-kinds allowing greater flexibility. The universal aspect of these type systems is also something that is seldom discussed.

We would argue that the types form an ontology of concepts, but can still be idiosyncratic: the “linguistic concepts”. As a general principle, when it comes to computational systems, the less information is needed in order to proceed with basic tasks (such as listing the compatibilities between lexical items), the better. We feel that having a type for every word in a given language (about 200,000) would result in a final system that is very demanding in terms of spacial complexity, as well as of Kolmogorov complexity (if each lexical entry is described in terms of words that are compatible with it). Getting closer to the selection classes of language would mean that the type system will have less redundancy, be smaller, and factorise common behaviours; a system of sub-typing for ontological inclusions is, obviously, a strong asset. However, it can be objected that we are not, at this point, able to produce a definitive system of sorts that comply with these objectives, while words have the advantage of being easy to list and ready to use, and certainly form a set of linguistic concepts.

While we certainly do not wish to claim that our assumptions are the only possible or valid type-theoretic formulations of lexical semantics, we wanted to explore the differences between the systems presented in order to understand what might be done in order to render them compatible. It is our belief that, even though the basic hypotheses are different, it is possible to encode any phenomenon presented in one framework using the syntax of any other.

## Summary

We presented a suitable relation of sub-typing for  $ATY_n$  that can be used with a system of types with many sorts and an organised lexicon to provide an analysis for different lexical phenomena from syntax to semantics. In order to provide a fully functional framework, acquiring the relevant sorts, lexical transformations, and ontological inclusions is the next step. Interesting perspectives for this have been provided by crowdsourced data in [Chatzikyriakidis et al., 2015] and enthymematic learning in [Breitholtz, 2015].

We also have discussed, using ontological inclusions and the common use of sub-typing relations in different frameworks, some of the differences between several type-theoretic systems modelling lexical semantics.

Despite those differences, we think that all of these systems share a common understanding of the difficulties involved by the adaptability of human language and the context-dependence of the lexicon, and of the possible type-theoretic mechanisms that can be used to integrate such complex phenomena in formal, compositional frameworks.

As discussed at many occasions, such as the TYTTLES workshop in ESSLLI Barcelona, August 2015, we believe that continued exchange of views between that growing community can result in one (or several) complete, functional, integrated frameworks that can successfully compete against big-data, statistical-based approaches to word sense disambiguation.

## References

- [Asher, 2011] Asher, N. (2011). *Lexical Meaning in Context: a Web of Words*. Cambridge University Press.
- [Asher and Luo, 2012] Asher, N. and Luo, Z. (2012). Formalization of coercions in lexical semantics. In Chemla, E., Homer, V., and Winterstein, G., editors, *Sinn und Bedeutung 17*, pages 63–80. <http://semanticsarchive.net/sub2012/>.
- [Bekki, 2014] Bekki, D. (2014). Dependent Type Semantics: An Introduction. In Christoff, Z., Galeazzi, P., Gierasimczuk, N., Marcoci, A., and Smet, S., editors, *Logic and Interactive Rationality (LIRa) Yearbook 2012*, volume I, pages 277–300. University of Amsterdam.
- [Bekki and Asher, 2012] Bekki, D. and Asher, N. (2012). Logical polysemy and subtyping. In Motomura, Y., Butler, A., and Bekki, D., editors, *JSAIL-isAI Workshops*, volume 7856 of *Lecture Notes in Computer Science*, pages 17–24. Springer.
- [Breitholtz, 2015] Breitholtz, E. (2015). Are Widows Always Wicked? Learning concepts through enthymematic reasoning. In Cooper, R. and Retoré, C., editors, *ESSLLI proceedings of the TYTTLES workshop on TYpe Theory and LEXical Semantics*, Barcelona.
- [Cardelli et al., 1991] Cardelli, L., Martini, S., Mitchell, J. C., and Scedrov, A. (1991). An extension of System-F with subtyping. In *Information and Computation*, pages 750–770. Springer-Verlag.
- [Chatzikyriakidis et al., 2015] Chatzikyriakidis, S., Lafourcade, M., Ramadier, L., and Zarrouk, M. (2015). Type Theories and Lexical Networks: Using Serious Games as the Basis for Multi-Sorted Typed Systems. In Cooper, R. and Retoré, C., editors, *ESSLLI proceedings of the TYTTLES workshop on TYpe Theory and LEXical Semantics*, Barcelona.

- [Cooper, 2007] Cooper, R. (2007). Copredication, dynamic generalized quantification and lexical innovation by coercion. In *Fourth International Workshop on Generative Approaches to the Lexicon*.
- [Cruse, 1986] Cruse, D. A. (1986). *Lexical Semantics*. Cambridge, New York.
- [Goddard and Wierzbicka, 2007] Goddard, C. and Wierzbicka, A. (2007). Semantic primes and cultural scripts in language learning and intercultural communication. In Sharifian, F. and Palmer, G. B., editors, *Applied Cultural Linguistics: Implications for second language learning and intercultural communication*, pages 105–124. John Benjamins.
- [Lang, 2014] Lang, F. (2014). Sous-typage coercif (coercive sub-typing). ENS Undergraduate Research Internship Report.
- [Longo et al., 2000] Longo, G., Milsted, K., and Soloviev, S. (2000). Coherence and transitivity of subtyping as entailment. *J. Log. Comput.*, 10(4):493–526.
- [Luo, 2011] Luo, Z. (2011). Contextual analysis of word meanings in type-theoretical semantics. In Pogodalla, S. and Prost, J.-P., editors, *Logical Aspects of Computational Linguistics*, volume 6736 of *Lecture Notes in Computer Science*, pages 159–174. Springer Berlin Heidelberg.
- [Luo, 2012] Luo, Z. (2012). Common nouns as types. In Béchet, D. and Dikovsky, A., editors, *Logical Aspects of Computational Linguistics*, volume 7351 of *Lecture Notes in Computer Science*, pages 173–185. Springer Berlin Heidelberg.
- [Luo et al., 2013] Luo, Z., Soloviev, S., and Xue, T. (2013). Coercive subtyping: Theory and implementation. *Inf. Comput.*, 223:18–42.
- [Mery et al., 2015] Mery, B., Moot, R., and Retoré, C. (2015). Typed Hilbert Operators for the Lexical Semantics of Singular and Plural Determiner Phrases. In *Epsilon 2015 – Hilbert’s Epsilon and Tau in Logic, Informatics and Linguistics*, Montpellier, France.
- [Mery and Retoré, 2013] Mery, B. and Retoré, C. (2013). Recent advances in the logical representation of lexical semantics. In *NCLS – Workshop on Natural Language and Computer Science, LiCS 2013*, Tulane University, New Orleans.
- [Mery and Retoré, 2013] Mery, B. and Retoré, C. (2013). Semantic Types, Lexical Sorts and Classifiers. In *NLPCS ’10- 10th International Workshop on Natural Language Processing and Computer Science - 2013*, Marseille, France.
- [Montague, 1974] Montague, R. (1974). The proper treatment of quantification in ordinary English. In Thomson, R. H., editor, *Formal Philosophy*, pages 188–221. Yale University Press, New Haven Connecticut.
- [Moot, 2010] Moot, R. (2010). Wide-coverage French syntax and semantics using Grail. In *Proceedings of Traitement Automatique des Langues Naturelles (TALN)*, Montreal.
- [Moravcsik, 1982] Moravcsik, J. M. (1982). How do words get their meanings ? *The Journal of Philosophy*, LXXVIII(1).
- [Nunberg, 1993] Nunberg, G. (1993). Transfers of meaning. In *Proceedings of the 31st annual meeting on Association for Computational Linguistics*, pages 191–192, Morristown, NJ, USA. Association for Computational Linguistics.
- [Pustejovsky, 1995] Pustejovsky, J. (1995). *The Generative Lexicon*. MIT Press.
- [Retoré, 2014] Retoré, C. (2014). The Montagovian Generative Lexicon  $\Lambda Ty_n$ : a Type Theoretical Framework for Natural Language Semantics. In Matthes, R. and Schubert, A., editors, *19th International Conference on Types for Proofs and Programs (TYPES 2013)*, pages 202–229.