On the Dual Interpretation of Nouns as Types and Predicates in Semantic Type Theories

William Babonnaud

On the Dual Interpretation of Nouns as Types and Predicates in Semantic Type Theories

William Babonnaud
Loria, Université de Lorraine, CNRS, Inria Nancy Grand-Est, Nancy, France
william.babonnaud@loria.fr

Abstract

In order to mediate the debate on whether common nouns are better interpreted as types or as predicates in type-theoretical semantic frameworks, the present paper shows that type theories whose models in category theory are *toposes* have access to a property drawing a one-to-one correspondence between first-order predicates and base types, thus enabling more flexibility for common noun interpretation. Using this flexibility and linguistic arguments based on negative predications, a subsequent proposal is made to interpret nouns as predicates with refined argument types.

1 Introduction

Any formal semantic framework can be characterised by its underlying type theory. One of the smallest possible theories for this purpose is Church's simple theory of types, built on two base types e and t for entities and truth values and an arrow constructor to create function types, which is characteristic of the grammar of Montague (1973). Yet the integration of lexical semantics into formal frameworks, motivated by the recognition of complex phenomena such as polysemy, selection restrictions and transfers of meaning (see e.g. Nunberg, 1979, 1995; Cruse, 1986), resulted in the use of richer type theories whose typical organisation consists in a subdivision of the type e into a hierarchy of *subtypes*, ordered by a subtyping relation, which intuitively enables the categorisation of entities according to some of their properties. In practice however, no consensus has been reached on what types should inhabit such a hierarchy. As listed by Retoré (2014), the collections proposed over the years vary from a set of around ten base types to a large system of one type for each common noun in the language.

This last position, initially introduced by Ranta (1994), have been more recently updated and defended in the works of Luo and Chatzikyriakidis (Luo, 2012a; Chatzikyriakidis and Luo, 2017), who argue that this so-called *CNs-as-types* approach provides a straightforward and efficient interpretation of common nouns (CNs). Under this view, the predication of a noun, e.g. *man*, on some entity x is represented as the typing judgement $x : man$. It is thus opposed to the more traditional and dominant *CNs-as-predicates* approach, which is directly inherited from Montague grammars and consists in interpreting *man* as a predicate $\mathbf{man} : e \to t$, in such a way that the same predication as above is interpreted as the logical formula $\mathbf{man}(x)$. Those alternative possibilities are actually parts of a larger conceptual shift from classical predicate calculus to systems based on Modern Type Theories (MTTs), among which appear Martin-Löf's type theory (Martin-Löf, 1984) and Luo's unified theory of dependent types (UTT) (Luo, 1994; Luo et al., 2013)[1].

Representing CNs as types rather than predicates follows the idea of distinguishing between nouns on the one side, and adjectives and verbs on the other. Types are thus interpreted as collections of entities, which then represents ranges of significance for propositional functions in a Russellian sense (see sect. 2.12 of Ranta, 1994). But what if such types could be defined as ranges of other functions? As suggested by Retoré (2014), some theories may indeed allow a dual interpretation, that is, a correspondence between types jugdements such as $x : man$ and truth on some predicate $\mathbf{man}(x)$. If Retoré is right, it could give access to a large amount of intermediate positions between CNs-as-types and CNs-as-predicates. This opens the debate on which way of interpreting CNs is the best-suited for natural language semantics, if any one is. Answering this question will have consequences on

[1] The various features that distinguish MTTs from Montague semantics will not be discussed here, see (Ranta, 1994) and (Luo, 2012a) for details.

the composition of the type hierarchy, as it varies in size and expressiveness depending on the choosen interpretation for CNs.

The present paper aims at mediating the debate between types and predicates by showing that, under the right settings, Retoré's suggestion—call it *CNs-as-both*—is an unavoidable property of some type theories, including UTT in its extension with coercive subtyping (Luo et al., 2013). As a result, the way of interpreting CNs and the type hierarchy are not constrained by the type theory itself, but would be a matter of lexicon design. We start defending this idea in Sect. 2 by highlighting the fact that the CNs-as-predicate approach stays relevant even in presence of subtyping, therefore nuancing Luo's position in (Luo, 2012b; Chatzikyriakidis and Luo, 2017). Then, in Sect. 3 we discuss the properties of UTT in light of its model in category theory, and establish that it has access to the CNs-as-both property. Finally, linguistic arguments to support an intermediate position between CNs-as-types and CNs-as-predicates, through the analysis of negation in sentences, is provided in Sect. 4.

2 Accomodating Predicates with Subtyping

In order to show that intermediate positions between types and predicates for CN interpretation are suitable, it is first necessary to examine the suitability of the CNs-as-predicate approach in the kind of type theories we are interested in. Recall that the type theories considered here are assumed to use a type hierarchy ordered by a subtyping relation. We will further assume the existence of a type constructor $\bullet$ to account for inherent polysemy (Pustejovsky, 1995), such that if a and b are base types, then the *dot type* $a \bullet b$ is considered a subtype of both a and b[2]. Hence, a predicate-based approach must be usable in presence of subtyping and dot types.

Yet its bad interaction with subtyping is one of the main criticisms addressed against the CNs-as-predicates approach, as demonstrated by Luo (2012b): consider the two types *phys* and *info* and the two words *book* and *heavy*, whose semantic interpretations would be a predicate $[\![\text{book}]\!] = \mathbf{book} : \mathit{phys} \bullet \mathit{info} \to t$ and a second-order predicate $[\![\text{heavy}]\!] : (\mathit{phys} \to t) \to (\mathit{phys} \to t)$. Then, the application of *heavy* on *book* shall raise on the semantic side the condition that *phys* must be a subtype of *phys* $\bullet$ *info*, which is the converse of the natural subtyping relation. Thus a simple application of an adjective to a CN in such a setting seems to be a rather complex task indeed. Luo concludes therefore that Montagovian grammars with their traditional CNs-as-predicates approach are unusable to deal with subtyping.

If the conclusion is right for Montagovian grammars, it has however to be nuanced when considering other frameworks of Montagovian inspiration, such as the Type Composition Logic (TCL) of Asher (2011) or the Montagovian Generative Lexicon (MGL) proposed by Retoré and his colleagues (Bassac et al., 2010; Retoré, 2014), as they often include additional strategies to overcome this subtyping problem in compositionality. In particular, we shall state that the demonstration above relies upon two hidden assumptions: (i) that direct application with subtyping is the only available operation for term composition, and (ii) that adjectives are necessarily interpreted as second-order predicates. The next paragraphs illustrate how new compositional strategies challenge these assumptions and enable the construction of subtyping-compliant frameworks.

Let us start with the functorial strategy proposed by Asher (2011) for TCL in cases where a dot type is involved, for instance when applying *heavy* to *book* as above. The intended meaning of the dot type *phys* $\bullet$ *info* is to represent two different aspects of *book*: one where the book is seen as a physical instance with a cover and pages, and another one involving only its informational contents. When combining *book* with *heavy*, we intend the latter to select only the physical aspect of the former, which licenses in a semantic framework a transformation of either $[\![\text{book}]\!]$ or $[\![\text{heavy}]\!]$—depending on the presuppositions to account for. TCL integrates these transformations as functors F and G which override subtyping constraints at application time by sending $[\![\text{heavy}]\!]$ to $F([\![\text{heavy}]\!]) : (\mathit{phys} \bullet \mathit{info} \to t) \to (\mathit{phys} \bullet \mathit{info} \to t)$ and **book** to $G(\mathbf{book}) : \mathit{phys} \to t$ respectively, thus enabling a more flexible account of term composition which still respects subtyping since the functors are restricted for use with dot types only[3].

[2] Note that this type constructor is definable in UTT with coercive subtyping as explained in (Luo, 2010).

[3] Actually, dot types are treated differently than subtyping in Asher's framework because the projections from the dot type to its aspects are not necessarily assimilable to subtyping relations from a theoretical point of view, see chap. 5 of (Asher, 2011) for discussion.

The implementation of such a strategy clearly rules out assumption (i), as it provides an alternative way of dealing with compositionality in particular cases. Another kind of compositional strategy, to be found e.g. in MGL, uses type polymorphism to get similar flexibility. We shall go further on this path by examining how polymorphism enables us to dismiss assumption (ii) as well. This assumption arises from the Montagovian tradition of seeing adjectives as noun modifiers, whence the second-order predicate interpretation given above. The semantic interpretation of *heavy*, for instance, is then obtained by a term of the following shape:

$$[\![\text{heavy}]\!] = \lambda P\, x.\, \textbf{heavy}(x) \wedge P(x) \qquad (1)$$

It involves a predicate **heavy** : $\mathit{phys} \to t$, which has the same type structure than the noun argument, and is embedded in a higher-order term with a logical conjunction to get the noun-modifier behaviour. Compared to other syntactical categories such as nouns or verbs, this interpretation of adjectives is singularly complex, not forgetting the difficulties it raises when interacting with subtyping. The semantic separation between nouns and adjectives is motivated by linguistic evidence that only nouns seem to bring support for quantification and counting, and the CNs-as-types approach as introduced by Ranta (1994) follows this idea. However, many grammarians and linguists have also pointed out the similitudes between these syntactical categories, thus supporting the possibility of treating nouns (or more accurately substantives) and adjectives as a continuum, where they would ultimately be distinguished by their predispositions with regard to linguistic functions and meaning specialisation (Jespersen, 1924; Guillaume, 1973; Gardelle, 2007).

If this continuum view is correct, it challenges the modern perception of adjectives as noun modifiers, not directly at the syntactic level where it stays relevant, but at the semantic one. Under the CNs-as-predicate approach, nouns are predicated of entities, and so could be adjectives. As a consequence, we could imagine a framework where the semantic interpretations of nouns and adjectives are of the same type shape. In the case of *heavy*, this means moving from the interpretation given in (1) above to its predicate component **heavy** : $\mathit{phys} \to t$ as the new term for $[\![\text{heavy}]\!]$. Composition would then be performed thanks to another polymorphic term, for instance in MGL or in the framework proposed by Babonnaud and de Groote (2020), which extends Montagovian grammars with records for modeling dot types, bounded polymorphism for composition, and a coercion-inference algorithm to account for subtyping. Assuming the integration of the latter framework in a syntax-semantic interface, the application of *heavy* to *book* would require to treat the deep syntactic relation between the adjective and the noun as the bounded polymorphic operator given in (2):

$$\Lambda\alpha.\lambda P\, Q\, x.P(x) \wedge Q(x) : \\ \forall \alpha < e.(\alpha \to t) \to (\alpha \to t) \to (\alpha \to t) \qquad (2)$$

The application of this operator to both $[\![\text{heavy}]\!]$ and $[\![\text{book}]\!]$ would trigger the inference algorithm, which would obtain a solution to the type constraints by unifying α with $\mathit{phys} \bullet \mathit{info}$ and introducing the coercion $c : \mathit{phys} \bullet \mathit{info} < \mathit{phys}$ to accomodate the variable to the predicate **heavy**. Thus, the resulting term would be $\lambda x.\textbf{heavy}(c(x)) \wedge \textbf{book}(x) : \mathit{phys} \bullet \mathit{info} \to t$.[4] Here again, the framework seems to be able to deal with subtyping while applying the CNs-as-predicates approach.

It should be noticed from the discussion in the two previous paragraphs that dismissing the assumption (ii) necessarily entails dismissing (i) as well, because traditional Montagovian grammars cannot support directly the combination of adjectives and nouns if they have the same type shape. Yet, the examples discussed above show that the addition of properties and features in frameworks of Montagovian inspiration enables them to accommodate the use of subtyping relation and dot types with the use of predicates. As a matter of fact, there is enough theoretical support to cope with the difficulties between subtyping and the CNs-as-predicates approach, which means that those difficulties cannot be an argument to definitively rule out the predicate view.

3 Theoretical Support to the Duality of Types and Predicates

3.1 On the Necessity of CNs-as-both

We now turn to the central question of this paper: is there any theoretical reason to choose one way

[4] It may be surprising in this approach that the resulting type is $\mathit{phys} \bullet \mathit{info} \to t$ and not $\mathit{phys} \to t$, since the former is not a subtype of the latter. This is not a problem for composition if we assume that other higher-order predicates also work with bounded polymorphic operators, as done by Babonnaud and de Groote (2020). Moreover, we may see such a type as coherent with the idea that *heavy book* still provides the possibility of copredication constructions.

of interpreting CNs rather than the other? Both CNs-as-types and CNs-as-predicates approaches have practical drawbacks—difficulties for modeling negation for the former, complex interaction with subtyping for the latter—and none of them is characteristic of a particular type theory: the Dependent Type Semantics of Bekki (2014) is an example of framework based on a MTT but interpreting CNs as predicates, and conversely it is not hard to conceive a Montagovian-style framework using types for CNs. The possibilities offered by the CNs-as-both property may enable the reconciliation of these views under a unified setting. In this section, we shall highlight that there actually are theoretical considerations supporting such a property in many frameworks.

Recall that having the CNs-as-both property means that for any entity x, we have for instance the typing judgement $x : \mathit{man}$ if and only if $\mathbf{man}(x)$ is true. Yet Chatzikyriakidis and Luo (2017) point out that such a definition in some settings (including simple type theory) may threaten the decidability of type checking. The reason lies in the "if" part of the equivalence: if one assert that the composition $\mathbf{man}(x)$ is well-typed, it results in $x : e$, and proving $x : \mathit{man}$ requires to prove the truth of the predications; but the truth of logical formulae is generally undecidable. However, Chatzikyriakidis and Luo exploit themselves the decidable direction of this duality—from types to predicates—for modeling negation: they introduce what they call a *predicational form* of typing judgement, that is, for any type, say *man*, a corresponding predicate $p_{man} : \mathit{man} \to t$ such that if $x : \mathit{man}$ then $p_{man}(x) = \texttt{true}$, where `true` is a tautological proposition[5]. In this case, the problem of type checking is avoided by the definition, since the well-typedness of $p_{man}(x)$ necessarily entails $x : \mathit{man}$.

The very fact that such a predicational form is needed even in a framework following the CNs-as-types approach is indicative of the practical interest that the CNs-as-both property could have. Yet we do not want to just define for each type a corresponding predicate which is true if and only if its argument is of the right type because it could threaten the decidability of some parts of a semantic analysis. Moreover, if we want such a property to be usable in a MTT-based framework, it would be better to provide a constructivist account of this duality. To establish such a result, we propose here to take a step forward in the path of type theory abstraction by studying which model UTT with coercive subtyping can receive in category theory.

3.2 A Categorical Perspective on UTT

To the best of the author's knowledge, computational linguistics have only made a sparse use of category theory and its results compared to other domains of computer science[6]. Through discussing the acceptability of the CNs-as-both property, we shall also illustrate how the categorical interpretation of type theories may help to design a framework suitable for natural language semantics. However, as the present paper cannot bring a full account of the definitions and properties useful to build a categorical model of UTT, we will stick to the fundamentals of category theory, and the curious reader is invited to consult other sources such as (Goldblatt, 1979; MacLane and Moerdijk, 1992; Johnstone, 2002) for more definitions and results.

A *category* is a collection of *objects* and, for each pair A, B of objects, a set of *morphisms* from A to B, obeying two additional laws. The convenient notation $f : A \to B$ is used to express the fact that f is a morphism from A to B; A and B are then respectively called *domain* and *codomain* of f. The additional laws are the following: first, there is an operation of composition on morphisms which sends $f : A \to B$ and $g : B \to C$ to the morphism $g \circ f : A \to C$ and is associative; and second, there is for any object A a morphism $id_A : A \to A$, called identity of A, which is neutral for right and left compositions. Two other categorical notions will be used in the rest of this paper: in any category, a *terminal object* is an object 1 such that for any object A there is a unique morphism $1_A : A \to 1$, and a *monomorphism* is a monorphism $f : A \to B$ such that for any object X and pair of morphisms $g, h : X \to A$, the equality $f \circ g = f \circ h$ implies $g = h$. We will use the notation $f : A \rightarrowtail B$ to indicate that f is a monomorphism from A to B. For illustration, a well-known example of category is **Set**, whose objects are all the possible sets and whose morphisms are functions between them, with the common definitions of composition and identity function. Furthermore

[5] For clarity and other reasons which will become clearer later in this section, we amalgamate the classical type of truth values and the intuitionistic type of propositions under the same notation *t*.

[6] Notable exceptions include (La Palme Reyes et al., 1994; Coecke et al., 2010; Asher, 2011).

the terminal objects of **Set** are the singletons, and its monomorphisms are exactly the injective functions[7].

A type theory may be interpreted as a category whose objects are types and morphisms are mappings between the corresponding types, enriched with additional properties as counterparts to each constructor or feature of the type system. Thus, a type a will correspond to an object A, a term of type $a \to b$ will be represented as a morphism $A \to B$ and, as a special case, an entity $x : a$ will correspond to a morphism $1 \to A$, assuming that a terminal object exists in the category[8]. Notable correspondences hold between cartesian closed categories and typed λ-calculus (Lambek, 1980), and between locally cartesian closed categories and Martin-Löf type theories (Seely, 1984)[9]. As for Luo's UTT with coercive subtyping, it can be captured by another kind of category called *topos*. Indeed, as explained by Luo (1994), the underlying logic of UTT is a higher-order one, and Lambek and Scott (1986) showed that the categories generated by such type theories are precisely toposes. Toposes already emerged in Asher's (2011) categorical model for TCL as suitable (and even necessary) for interpreting dot types, and Babonnaud (2019) further argues that toposes could be the best categorical models to interpret on a unified basis a large variety of semantic frameworks with subtyping.

The definition of a topos includes several key properties that will not be exhaustively listed here[10]. For our purposes, it is enough to know that the relevance of toposes for semantic models comes from the existence in these categories of a particular object Ω called *subobject classifier*, which can be interpreted as the categorical counterpart for the type t of truth values or propositions—depending on the underlying logic of the chosen type system. The subobject classifier is characterised by a specific morphism $\top : 1 \to \Omega$ which represent the value *true* (or the tautological proposition `true`), and a universal property which binds it to the existence of monomorphisms which, as argued by Babonnaud (2019), can interpret subtyping relations. This property is formally embodied into the following Ω-axiom (Goldblatt, 1979):

Ω-axiom. *For every monomorphism $f : B \rightarrowtail A$ there is a unique morphism $\chi_f : A \to \Omega$ such that:*

(i) $\chi_f \circ f = \top \circ 1_B$*; and*

(ii) for any object C and morphism $g : C \to A$ such that $\chi_f \circ g = \top \circ 1_C$, there is a unique morphism $h : C \to B$ such that $f \circ h = g$.[11]

This axiom, conjointly with the property of toposes to have all finite limits (see Goldblatt, 1979; MacLane and Moerdijk, 1992; Johnstone, 2002), have an important consequence once transposed in a semantic type system. Assume a topos $\mathcal{T}$ with a distinguished object E corresponding to e. A first-order predicate such as **man** $: e \to t$ is interpreted as a morphism $man : E \to \Omega$. The properties of $\mathcal{T}$ ensure that there exists an object M along with a monomorphism $c : M \rightarrowtail E$[12] such that $man \circ c = \top \circ 1_M$, that is, there exists a type *man* in the system such that $x :$ *man* entails **man**$(c(x)) =$ `true`. Conversely, one can also prove that if $y : e$ is such that **man**$(y) =$ `true`, then there is an $x :$ *man* such that $y = c(x)$[13]. As a result, the Ω-axiom is a theoretical support in toposes for the duality between predicates and types, and as the categorical model generated by UTT is a topos, we conclude that this axiom and its consequences are also part of UTT.

3.3 Translation of Topos Properties into UTT

To know that the Ω-axiom exists in UTT is one thing, but understanding how this axiom manifests itself is another one which shall be clarified here. But before exploring in more details its practical consequences in the theory, let us give a few words about the predicational form of typing judgement proposed by Chatzikyriakidis and Luo (2017). It is

[7] Note that not all categories may have terminal objects and monomorphisms. Moreover, this example shows that terminal objects are defined up to isomophism, so that the notation 1 and reference to *the* terminal object of a category are valid by misuse of language.

[8] The type-theoretical counterpart of the terminal object 1 is generally the unit type.

[9] The reader may consult (Bell, 2012) among others for history and details on categories and their equivalence with type theories.

[10] While not directly linked to the present discussion, it may be worth noticing at least that toposes have all the properties of the kinds of categories mentioned above, that is, cartesian closed and locally cartesian closed categories. As a consequence, a topos is also acceptable as a model for simple type and Martin-Löf's theories.

[11] This also entails $1_B \circ h = 1_C$, but it is already true by definition of the terminal object. Category theorists shall recognise in this property the defintion of B as the *pullback* of χ_f and $\top$.

[12] The object M is then said to be a *subobject* of E, whence the name of *subobject classifier* given to Ω.

[13] It is so because variables of type e are interpreted as morphisms $1 \to E$. The result then comes from direct application of property (ii) of the Ω-axiom

not hard to see that the corresponding morphism of p_a in the topos model $\mathcal{T}$ of UTT is $\top \circ 1_A : A \to \Omega$: not only this morphism has the expected domain and codomain, but if we take a morphism $x : 1 \to A$ corresponding to an entity of type a, we can also prove that $\top \circ 1_A \circ x = \top$, which is the desired property[14]. Then we notice that this morphism $\top \circ 1_A$ is exactly the kind of morphism that appears in the right-hand side of the equalities in the Ω-axiom. As a consequence, an interpretation of the axiom is the following: if we are given a subtyping coercion $c : a \leq b$, then we can find a predicate $\chi_c : b \to t$ such that $\chi_c \circ c = p_a$.

But what kind of predicate would be χ_c? Can we define it in a constructivist way? Given that UTT, as shown by Luo (1994), allows for building higher-order propositions, we can answer by the affirmative. Assume a constant $c : a \to b$ corresponding to the coercive subtyping $a \leq b$. Then, pose the following:

$$\chi_c := \lambda y : b.\, \exists x : a.\, (c(x) = y) : b \to t \qquad (3)$$

The existentially quantified part of this term is a definable proposition in UTT. Thus χ_c is defined as the "characteristic" of a in b, such that χ_c is true on y if and only if y is in the image of the coercion function c. This shows how the conversion from type to predicate works in such type theories. As for the reverse direction of the type-predicate duality, if we are given any predicate $P : b \to t$, then following Seely (1984) the corresponding type may be defined as:

$$a := \Sigma x : b.\, (P(x) = \texttt{true}) \qquad (4)$$

where Σ denotes a dependent sum[15]. In other words, a as defined in (4) is the type of pairs (x, p) where $x : b$ and p is a proof that $P(x)$ is true. As declared by Luo (2010), the first projection π_1 of such a sum is indeed a subtyping coercion, that is, $\pi_1 : a \leq b$ as required.

The formulations in (3) and (4) are sound in the sense that if $c : a \leq b$ is a coercion then we have a proof of $x : a$ if and only if we have a proof of $x : \Sigma y : b.(\chi_c(y) = \texttt{true})$, and if $P : b \to t$ is a predicate and $\pi : (\Sigma x : b.P(x) = \texttt{true}) \to b$ is the corresponding subtype coercion, we have a proof of $P(x) = \texttt{true}$ if and only if we have a proof of $\chi_\pi(x) = \texttt{true}$. Hence there is a way to move from types to predicates: for instance, if $x : \mathit{man}$ and $c : \mathit{man} \leq e$, it suffices to define $\mathbf{man} = \lambda x.\exists y.(c(y) = x) : e \to t$ to have a corresponding predicate with $\mathbf{man}(x) = \texttt{true}$.

We should however point out that the categorical model presented in this paper, as well as the properties described, do not suffice to ensure that the corresponding type theory has key properties such as normalisation and decidability, nor that its implementation would be easier[16]. Moreover, as Chatzikyriakidis and Luo (2017) warned, we still have a threat to type-checking decidability when trying to move from predicates to types since proving $P(x) = \texttt{true}$ in a MTT may be as hard as proving the truth of $P(x)$ in classical predicate calculus. To overcome this difficulty, a solution could be to adapt the type system so that the types of predicate arguments are themselves subtypes of e in order to make type checking more precise, as shall be discussed in the next section.

4 A Linguistic Perspective on Base Types

4.1 Type Theory and the Lexicon

Let us start with the following observation: if a is the type interpretation of some CN, then any b such that $a \leq b \leq e$ defines a possible predicational interpretation of this CN, the extreme cases being respectively Chatzikyriakidis and Luo's predicational form $p_a : a \to t$, and regular Montagovian predicates $e \to t$. A topos type theory therefore provides a full range of predicate interpretations for CNs which only depends on the types we accept in the hierarchy between their direct type interpretations and the greatest base type e. Besides, we shall notice that the Ω-axiom applies to *any* predicate, which may include for instance adjectives and intransitive verbs: we may then obtain some unexpected types such as the type *heavy* of heavy entities. Hence we may end up with an unreasonable amount of types and a large variety of possible interpretations for predicates—on top of the type checking difficulty.

[14] The complete proof runs as follows: by associativity of the composition, $(\top \circ 1_A) \circ x = \top \circ (1_A \circ x)$, and $1_A \circ x$ is a morphism $1 \to 1$. Yet, by definition of 1, there can be only one morphism $1 \to 1$, which is its identity id_1. Hence, $\top \circ (1_A \circ x) = \top \circ id_1 = \top$ because the identity is neutral for composition.

[15] As mentioned in footnote 11, this works because by Ω-axiom the object A corresponding to a is the pullback of $P : B \to \Omega$ and $\top$.

[16] As an anonymous rewiever rightly highlighted, the categorical interpretation of dependent types of Seely (1984) suffered from coherence issues that may be treated by a careful work on models, has discussed e.g. in (Curien et al., 2014).

We claim that the key solution for this issue is a proper definition of the *lexicon*, whose idea follows the lines of Pustejovsky and Batiukova (2019): a mapping between linguistic lemmas and their semantic representation, along with all relevant data for compositionality. It manifested earlier in this paper as the bracket application $[\![.]\!]$ sending a word to its typed interpretation. Facing the profusion of possible interpretations of a word in our type theory, a carefully-designed lexicon behaves as a filter which selects one (sometimes two) interpretation to be used. Thus we are not committed to use all the theoretically possible types and predicates, but only a proper subset thereof—without ruling out the properties of the type system. In other words, the lexicon may provide the basis to define a decidable fragment of a type theory by restriction to the terms using the constants it carries, without need to apply the type-predicate duality anymore within that fragment.

This puts a new highlight on the question of what types inhabit the hierarchy: the problem does not lie in the type theory proper, but rather at the interface between language and semantic representation, that is, in the lexicon. The main concern of lexicon design is therefore to choose types and predicates that are relevant for distinguishing between straightforward cases of semantic composition and other phenomena such as coercions, and this choice should obey the following criterion: types must be precise enough to distinguish the various cases of composition, but not too precise if this precision is not relevant for compositional matters. As such, we intend to separate the notion of type from the notion of meaning to the extent that the involvement of types in a semantic analysis is narrowed to the compositional behaviour of words, abstracting upon the other dissimilarities they may have.

4.2 Types in Compositionality and Negative Predications

To explain this latter idea, consider for instance the words *cat* and *dog* and their corresponding interpretations under the CNs-as-types approach. What does distinguish them from a compositional point of view? As a matter of fact, both words are very similar to that extent: lots of predicates which are meaningful on one are also meaningful on the other, as to be seen with physical descriptions, actions or even moods. Only a few words seem to resist such an analysis, among which the verbs corresponding to their respective cries, *meow* and *bark*. Yet even the compositional power of these predicates w.r.t. *cat* and *dog* is questionable: how meaningless is it to apply e.g. *meow* to *dog*? We are prone to think that a meowing dog is an absurdity, and that *meow* should be only employed with cats, hence a typing restriction $\mathbf{meow} : cat \to t$. Let however constrast this view with the sentences in (1) below:

(1) a. Dogs do not meow but bark.
b. #Tables do not meow but bark.

The sentence (1a) is obviously true, while (1b) is anomalous. Now, if we forget about the second verb in both sentences, we have to admit that the resulting ones should still differ in meaning, since tables and dogs fail to meow for different reasons. Moreover, reversing the verbs would turn (1a) to falsity but would keep (1b) anomalous, which hints that a meaningful predication of *meow* to *dog* could be possible. As a consequence, we shall recognise that the negative predications in (1a) and (1b) have different underlying logics.

Actually, as discussed by Horn (1989), many philosophers have recognised at least two forms of negation which, following the terminology of Sommers (1965, 1971), will be called here *negation* and *denial*. The subtle difference between them can be illustrated by the sentences in (2), where it is integrated in the distinction between the use of *not* for negation versus the prefix *un-* for denial, and results in a divergence in semantic acceptability:

(2) a. Triangles are not intelligent.
b. #Triangles are unintelligent.

Their distinction is a matter of application level and presupposition: denial applies to predicates so that in a meaningful predication either the predicate or its denial is true on the argument, while negation applies directly to propositions and do not obey this excluded middle clause. Thus (2b) is anomalous because triangles can be neither intelligent nor unintelligent, hence a meaningless predication. By constrast, (2a) is acceptable under the reading which states precisely that *intelligent* (or its denial) cannot apply to triangles.

However, for predicates like *bark* and *meow* both negation and denial are rendered by the particle *not*, leading to an ambiguity in the negative predications in (1). Yet those sentences show that these predicates belong to the same "meaning scale" where they contrast each other, similarly to the opposi-

tion between *intelligent* and *unintelligent*. We may reasonably assume that all predicates from such a meaning scale are meaningful on the same arguments; from which we may conclude that (1a) contains a denial, while (1b) contains a pure negation. As a result, *meow* applies meaningfully to *dog* and meaninglessly to *table*, a distinction capturable with semantic types by extending the span of *meow* from *cat* to a greater argument type which includes *dog* and *cat*, but excludes *table*. If the type *animate* fits such a role, then we end up with a new interpretation $[\![\text{meow}]\!] = \textbf{meow} : \textit{animate} \to t$ in the lexicon.

The analysis above, if correct, should be reproduceable on any predicate which seems to distinguish *cat* and *dog* from a compositional point of view, including the nouns themselves in their predicative use if we accept that sentences in the sort of *"cats are not dogs"* are actually denials. From this generalisation, we conclude that the types *cat* and *dog* are not needed in our lexicon, since all predicates are blind to the distinction they introduce w.r.t. compositionality. Their interpretations, instead of relying on those types, would use the supertype *animate* in predicates $\textbf{cat} : \textit{animate} \to t$ and $\textbf{dog} : \textit{animate} \to t$, in such a way that any entity determined to be a dog or a cat would receive by type-checking the type *animate* in question, which is sufficient for further predications. Any other kind of difference between cats and dogs is not a matter of compositionality anymore, and should rather be accounted for in deeper semantic or pragmatic analyses.

4.3 Interpreting Negation in Type Theories

The previous discussion raises the issue of interpreting negation in semantic type theories. Chatzikyriakidis and Luo (2017) propose a polymorphic operator NOT rather than the usual connective $\neg$ to ensure consistency with their predicational forms[17], since well-typedness of $\neg p_{man}(x)$ enforces the contradictory condition $x : \textit{man}$ whereas $\text{NOT}(p_{man}, x)$ does not. However, this interpretation is not sufficient to distinguish negation from denial because predicational forms are too restrictive: in particular, $\text{NOT}(p_{dog}, x)$ has the same meaning regardless of x being of type *cat* or *table*.

Taking a broader predicate interpretation for *dog* may solve this problem as well-typedness of $\neg\textbf{dog}(x)$ holds for $x : \textit{animate}$, and $\textit{cat} < \textit{animate}$. We may thus interpret denials using $\neg$, and keep the operator NOT for negations, so that $\text{NOT}(\textbf{dog}, x)$ would mean that the type of x is uncompatible with the argument type of **dog**. It appears then that NOT enables us to transpose the type-theoretic property of type incompatibility into a logical formula. Another option for negation could be to use the most general predicate $\textbf{dog}' : e \to t$ while using **dog** only for denials, replacing $\text{NOT}(\textbf{dog}, x)$ by $\neg\textbf{dog}'(k(x))$ with k an adequate coercion to e, but that would require an additional mechanism to introduce such a predicate when needed, for we cannot be committed to have this general interpretation available in the lexicon.

5 Conclusion

A first observation about the interpretation of CNs is the fact that neither types nor predicates seem to offer a greater practical advantage: in both cases, adding little theoretical machinery into the framework enables their interaction with subtyping and dot types in a fairly straightforward way. In the case of predicates however, this interaction comes at the cost of reconsidering composition, as it generally requires new mechanisms that go beyond direct application. Nevertheless, such a revision in compositionality may be an unavoidable step towards better interpretations of complex semantic phenomena anyways.

Yet the main observation of the present paper is the fact that any type theory with enough assumptions can actually model both views of interpreting CNs in a equivalent way, by establishing a bijective correspondence between predicates of type $e \to t$ and subtypes of e. Theories with this property include MTTs such as Martin-Löf type theory and Luo's UTT, and further considerations show that many other type-theoretical frameworks can be extended to get access to this property as well. The discussion in Sect. 3 shows how abstracting type theories through the perspective of category theory helps in identifying and establishing their key properties. The present paper particularly highlighted the correspondence between the expected duality of interpretations and a general property of the categorical class of *toposes*.

[17] In UTT, the operator NOT has the polymorphic type $\Pi\alpha : \text{CN}.(\alpha \to t) \to (e \to t)$ (by assimilating propositions to t and objects to e), where CN is the type universe of CN types. Note that this universe seems to correspond conceptually to the collections of subtypes of e. As a consequence, types of the form $\Pi\alpha : \text{CN}.\tau$ and of the bounded polymorphic form $\forall\alpha < e.\tau$ from (Babonnaud and de Groote, 2020), as used in Sect. 2, may be seen as equivalent.

This duality gives access to intermediate choices of interpretation for CNs which blur the lines between the strict applications of each view. In Sect. 4, we proposed an interpretation of CNs as predicates with refined argument types, whose choice relies on their compositional abilities with other predicates from the language, and the possible arguments thereof. This refinement requires to determine on which entities it is meaningful to assert or deny the given predicate, thus excluding the entities on which the predication is absurd. In negative sentences, this amounts to be able to dissociate pure negation and denials. Generalising such a reasoning on a natural language should eventually lead to the construction of a lexicon using a proper sub-hierarchy of the possible types, each type corresponding to some cluster of CNs with the same compositional behaviour.

References

Nicholas Asher. 2011. *Lexical Meaning in Context: A Web of Words*. Cambridge University Press.

William Babonnaud. 2019. A topos-based approach to building language ontologies. In *Formal Grammar. 24th International Conference, FG 2019, Riga, Latvia, August 11, 2019, Proceedings*, pages 18–34, Berlin. Springer.

William Babonnaud and Philippe de Groote. 2020. Lexical selection, coercion, and record types. In *LENLS17: Logic & Engineering of Natural Language Semantics, Online, November 15-17, 2020*.

Christian Bassac, Bruno Mery, and Christian Retoré. 2010. Towards a type-theoretical account of lexical semantics. *Journal of Logic, Language, and Information*, 19(2):229–245.

Daisuke Bekki. 2014. Representing anaphora with dependent types. In *Logical Aspects of Computational Linguistics. 8th International Conference, LACL 2014, Toulouse, France, June 18-20, 2014. Proceedings*, pages 14–29, Berlin. Springer.

John L. Bell. 2012. Types, sets, and categories. In Dov M. Gabbay, Akihiro Kanamori, and John Woods, editors, *Sets and Extensions in the Twentieth Century*, volume 6 of *Handbook of the History of Logic*, pages 633–687. North-Holland Publishings.

Stergios Chatzikyriakidis and Zhaohui Luo. 2017. On the interpretation of common nouns: Types versus predicates. In Stergios Chatzikyriakidis and Zhaohui Luo, editors, *Modern Perspectives in Type-Theoretical Semantics*, volume 98 of *Studies in Linguistics and Philosophy*, pages 43–70. Springer.

Bob Coecke, Mehrnoosh Sadrzadeh, and Stephen Clark. 2010. Mathematical foundations for a compositional distributional model of meaning. *Linguistic Analysis*, 36:345–384.

David A. Cruse. 1986. *Lexical Semantics*. Cambridge University Press.

Pierre-Louis Curien, Richard Garner, and Martin Hofmann. 2014. Revisiting the categorical interpretation of dependent type theory. *Theoretical Computer Science*, 546:99–119.

Laure Gardelle. 2007. Adjectifs dits 'substantivés' et noms composés : quel continuum entre adjectif et nom ? In *Journée d'études concours – Université Jean Moulin Lyon III*, Lyon, France.

Robert Goldblatt. 1979. *Topoi: The Categorial Analysis of Logic*, volume 98 of *Studies in logic and the foundations of mathematics*. North-Holland Publishings.

Gustave Guillaume. 1973. *Principes de linguistique théorique*. Les Presses de l'Université Laval, Québec.

Laurence R. Horn. 1989. *A Natural History of Negation*. The University of Chicago Press.

Otto Jespersen. 1924. *The Philosophy of Grammar*. George Allen & Unwin Ltd., London.

Peter T. Johnstone. 2002. *Sketches of an Elephant: A Topos Theory Compendium*. Oxford University Press.

Marie La Palme Reyes, John Macnamara, and Gonzalos E. Reyes. 1994. Reference, kinds and predicates. In John Macnamara and Gonzalo E. Reyes, editors, *The Logical Foundations of Cognition*, volume 4 of *Vancouver Studies in Cognitive Science*, pages 91–143. Oxford University Press.

Joachim Lambek. 1980. From λ-calculus to cartesian closed categories. In J. Roger Hindley and Jonathan P. Seldin, editors, *To H.B. Curry: Essays on Combinatory Logic, Lambda Calculus and Formalism*, pages 375–402. Academic Press, London.

Joachim Lambek and Philip J. Scott. 1986. *Introduction to Higher Order Categorical Logic*. Cambridge University Press.

Zhaohui Luo. 1994. *Computation and Reasoning: A Type Theory for Computer Science*. Oxford University Press.

Zhaohui Luo. 2010. Type-theoretical semantics with coercive subtyping. In *Proceedings of SALT 20*, pages 38–56.

Zhaohui Luo. 2012a. Common nouns as types. In *Logical Aspects of Computational Linguistics. 7th International Conference, LACL 2012, Nantes, France, July 2-4, 2012, Proceedings*, pages 173–185, Berlin. Springer.

Zhaohui Luo. 2012b. Formal semantics in modern type theories with coercive subtyping. *Linguistics and Philosophy*, 35(6):491–513.

Zhaohui Luo, Sergei Soloviev, and Tao Xue. 2013. Coercive subtyping: Theory and implementation. *Information and Computation*, 223:18–42.

Saunders MacLane and Ieke Moerdijk. 1992. *Sheaves in Geometry and Logic: A First Introduction to Topos Theory*. Springer, New York. Corr. 2nd edition 1994.

Per Martin-Löf. 1984. *Intuitionistic type theory: Notes by Giovanni Sambin of a series of lectures given in Padua, June 1980*. Bibliopolis, Napoli.

Richard Montague. 1973. The proper treatment of quantification in ordinary english. In Patrick Suppes, Julius Moravcsik, and Jaakko Hintikka, editors, *Approaches to Natural Language*, pages 221–242. Reidel, Dordrecht.

Geoffrey Nunberg. 1979. The non-uniqueness of semantic solutions: Polysemy. *Linguistics and Philosophy*, 3(2):143–184.

Geoffrey Nunberg. 1995. Transfers of meaning. *Journal of Semantics*, 12(2):109–132.

James Pustejovsky. 1995. *The Generative Lexicon*. MIT Press, Cambridge.

James Pustejovsky and Olga Batiukova. 2019. *The Lexicon*. Cambridge University Press.

Aarne Ranta. 1994. *Type-Theoretical Grammar*. Oxford University Press.

Christian Retoré. 2014. The montagovian generative lexicon ΛTy_n: a type theoretical framework for natural language semantics. In *Proceedings of the 19th International Conference on Types for Proofs and Programs*, volume 26 of *LIPICS*, pages 202–229.

Robert A. G. Seely. 1984. Locally cartesian closed categories and type theory. *Mathematical Proceedings of the Cambridge Philosophical Society*, 95(1):33–48.

Fred Sommers. 1965. Predicability. In Max Black, editor, *Philosophy in America*, pages 262–281. Cornell University Press, Ithaca.

Fred Sommers. 1971. Structural ontology. *Philosophia*, 1:21–42.