Selectional Restrictions, Types and Categories

Nicholas Asher

Selectional restrictions, types and categories ☆

Nicholas Asher

CNRS, Laboratoire IRIT, UMR 5505, Université Paul Sabatier, Toulouse, France

A B S T R A C T

The expressions of a language distinguish between many different types of objects. These types can affect how the meanings of these expressions combine. This paper provides a formal picture of the process of meaning combination in a richly typed framework.

1. Introduction

Natural language countenances many different types of objects that can affect predication. Here are some that linguists and computer scientists make use of either in formal semantics or in designing applications for the semantic web. The *vs.* indicates that the types are viewed as incompatible.

(1) count vs. mass; places vs. portions of matter; kind vs. individual; abstract vs. concrete (informational object vs. physical object); eventualities vs. objects; animate vs. non-animate; propositions vs. facts vs. eventualities; locations vs. objects.

Some of these types are distinguished by the predicates of which they can be arguments. For example, *person* is a paradigm count noun, and *blood* is a paradigm mass noun; the determiners that they can combine with form two almost disjoint sets

(2) a person, #a blood.
(3) #much person, much blood.

Linguists have also devised other tests like the copredication test to see whether types are really distinct or not. Copredications involve applying two different arguments to the same predicate in a coordinate construction. If the types do not matter for linguistic applications, then the copredication should succeed, as in (4c). But if the copredication is infelicitous or zeugmatic, then this points to a difference the distinct types make in the composition of meaning. In (4), I provide evidence that places and portions of matter make a difference to semantic composition.

(4) a. John swept the kitchen.
b. John swept the dust.
c. John swept the dust and the leaves.
d. #John swept the dust and the kitchen.

This paper answers two questions: how do these types make a difference to predication, and how do we integrate a rich system of types into semantic composition?

☆ Many thanks to Christian Rétoré and to anonymous reviewers of the special issue for helpful comments.
E-mail address: Nicholas.Asher@irit.fr.

2. Not all predications are equal

Syntax has a good deal to say about semantic well-formedness. You can't predicate a finite verb of another finite verb as in *likes sat*; it's unintelligible. But sometimes predications also go wrong for semantic reasons, because the types of the arguments don't fit with the types demanded by their predicates. Most linguists assume that words that can be used as predicates impose what they call *selectional restrictions* on what they take as arguments. When those restrictions are incompatible with the type of the argument, we get a semantic clash.

(5) a. ?That person contains an interesting idea about Freud.
b. That person has an interesting idea about Freud.
c. That book contains an interesting idea about Freud.
d. That person is eating breakfast.
e. That book is red.
f. #That rumor is red.
g. #The number two is red.
h. #The number two is soft.
i. #The number two hit Bill.
j. The number two is prime.
k. John knows which number to call.
l. *John believes which number to call.

The predications in (5f,g,h) or (5i) are malformed—each contains what Gilbert Ryle would have called a category mistake. For [19], most natural language expressions come with restrictions on what other words they can combine with; to apply a predicate to a term that does not meet those restrictions results in a category mistake. Numbers as abstract objects can't have colors or textures or hit people; it's nonsensical in a normal conversation to say something like the number two is red, soft, or that it hit Bill. If these sentences are admissible it is only because the noun phrase *the number two* does not denote the abstract object, but is coerced to some other denotation.[1] The mismatch between predicate and argument is even more blatant in (5l).

One has to exercise some care in understanding why a predication like (5a) sounds so much odder than (5b–d). In some sense people can contain information: spies have information that they give to their governments and that counter-spies want to elicit; teachers have information that they impart to their students. But one can't use the form of words in (5a) to straightforwardly convey these ideas. The predication is odd; it involves a misuse of the word *contain*. If it succeeds at all in making sense to the listener, it must be subject to reinterpretation.

It's important to distinguish between necessary falsity and the sort of semantic anomaly present in (5a) and (5f–i). In the history of mathematics, many people, including famous mathematicians, have believed necessarily false things. But competent speakers of a language do not believe propositions expressed by a sentence with a semantically anomalous predication. (5a) or (6c,d) are semantically anomalous in a way that (5b–d) or (6a,b) below are not.[2]

(6) a. Tigers are animals.
b. Tigers are robots.
c. #Tigers are financial institutions.
d. #Tigers are Zermelo–Frankel sets.

Many philosophers take (6a) to be necessarily true and (6b) to be necessarily false.[3] Nevertheless, according to most people's intuitions, a competent speaker could entertain or even believe that tigers are robots; he or she could go about trying to figure this out (e.g., by dissecting a tiger). It is much harder to accept the possibility, or even to make sense of, a competent speaker's believing or even entertaining that tigers are literally financial institutions, let alone ZF style sets. Thinking about whether a competent speaker could entertain or believe the proposition expressed by a sentence gives us another means to distinguish between those sentences containing semantically anomalous expressions and those that do not.

Given these considerations, it seems that natural languages take selectional restrictions seriously and an account of semantic composition must give an account of these. It's also clear that the information relevant to selectional restrictions is much more detailed and finegrained than the types considered in Montague Grammar, for instance. Montague grammar has only two primitive types, the type of entities e and the type of truth values t and TY2 only adds the type of possible worlds s [6]. Clearly, we cannot identify the requirements provided by selectional restrictions with type requirements, if types are understood à la Montague or TY2.

[1] As the attentive reader may have already guessed, besides "normal" conversations, there are also "abnormal" discourse contexts—contexts that would enable us to understand these odd sentences in some metaphorical or indirect way, or that even enable us to reset the types of words. I won't have anything to say about this here.

[2] Thanks to Dan Korman for the first two examples.

[3] The reason for this has to do with a widely accepted semantics of natural kinds due to Hilary Putnam and Saul Kripke, according to which *tigers* picks out a non-artifactual species in every possible world.

Most linguists and computer scientists interested in lexical semantics take selectional restrictions to be *sortal* restrictions. Let's call such an approach the Sortal Theory; it is enshrined in most description logics. The Sortal Theory claims that type information is really just a part of the language of logical forms that is singled out by selectional restrictions. Type information is really just information about sorts. Semantic ill-formedness is the result of using the semantic and proof theoretic apparatus of the language of logical forms to derive a contradiction based on meaning postulates about sorts.

At first glance, the distinction between the Sortal Theory and a typed theory of logical forms looks to be one of notational variance. The typed lambda term of the theory I will propose here in (7a) (and have defended in [1]) has the very similar looking Sortal Theory analogue in (7b) ($\phi_\alpha(x)$ is the object level formula capturing the information associated with the type assignment $x{:}\,\alpha$).

(7) a. $\lambda x{:}\ \alpha\psi(x)$.
b. $\lambda x(\psi(x) \wedge \phi_\alpha(x))$.

But here appearances deceive. Sorts, as predicates in logical form, have a set theoretic semantics; and they provide no constraints on the construction of logical form. Without any types at all, we have all the inconveniences of doing compositional semantics in a type-free environment. Furthermore, without types, we have no account of semantic ill-formedness, which my theory accounts for in terms of the failure of the type presuppositions of terms to be justified and hence for a failure of normalization. According to the Sortal Theory, there aren't any syntactically well-formed sentences that are semantically ill-formed; those sentences that are predicted to be ill-formed because of type clashes on the typed view are just false on the Sortal Theory. The Sortal theory also just seems to get the truth conditions of simple sentences wrong, once they are embedded under truth functional operators. For purposes of illustration, let's assume that *hit*, when we translate it into a λ term imposes a type restriction PHYSICAL-OBJECT or P on its arguments. According to the Sortal Theory,

(8) John didn't hit Mary $\rightsquigarrow$
$\neg\mathrm{P}(j) \vee \neg\mathrm{P}(m) \vee \neg H(j,m)$.

(8) is true in any world in which Mary or John are not physical objects. These are obviously not the right truth conditions for such a sentence. We don't want the type information given in selectional restrictions to be part of the proferred content of statements. The Sortal Theory is wrong, because it does just this.

I propose that indeed that selectional restrictions be treated as type information, and that sortal restriction be conceived of as a sort of type checking on semantic well-formedness. But first, we must see how what sort of information sortal restrictions really provide.

3. Type presuppositions

In the last section we saw evidence that selectional restrictions don't enter into proferred content, the content that is the focus of attention of what is said and that interacts with the scope of various truth functional and other operators like negation or modality. Instead they seem to project outside of proferred content; that is, their content lies outside the scope of negation or modality, even when these operators seem to take sentence wide scope. Selectional restrictions function similarly to what linguists call presupposed content. Presuppositions are largely defined in terms of their projection behavior. In (9a–d), the presupposed content of definite descriptions projects outside the scope of modal, truth functional operators and even questions. Linguists claim that if the presuppositional content cannot be added to the global content at that point, the statements in (9) should sound odd. On the other hand, linguists say that presupposed content can be *bound* by proferred content that entails it. In that case, the presuppositions don't project as in (9e).

(9) a. The present King of France could have been bald.
b. Is the present King of France bald?
c. The present King of France is not bald.
d. If Sylvain's son is three years old, then he must be going to school.
e. If Sylvain has a son, then Sylvain's son is almost three years old.

We note a similar behavior with selectional restrictions. The oddness of trying to predicate a physical property of an abstract object persists under embeddings. There is also a phenomenon similar to binding as in (10d). More interestingly, it seems as though proferred content as in (10e) can override and reset standard type presuppositions.

(10) a. ?The number two could have been red.
b. ?Is the number two soft?
c. ?The number two didn't hit Bill.
d. If Bill were a natural number, then he would be either prime or composite.
e. If numbers were physical objects, then the number two could have been red.

There are also other tests for presuppositions that selectional restrictions meet. The first test involves the non-redundancy of presupposed content. Making presuppositions an explicit part of proferred content does not lead to redundancy when a term triggering that presupposition is subsequently used in a discourse context. Type presuppositions have this feature as well.

(11) Two is an abstract object. Two is prime.
(12) Sylvain has a son. Sylvain's son is almost three years old.

The second test involves the inability to make certain discourse continuations on presupposed content as in (13). This correlates with the impossibility of making discourse continuations on type requirements

(13) After Susan got mad at John, she made up with him. #She (had) slapped him.

I will henceforth take selectional restrictions as type presuppositions, a view I defend at greater length in [1].

The presupposition terminology is quite relevant to determining the role of selectional restrictions in semantic composition. From the expansive literature on type presuppositions[4] in linguistics, it is well-known that different lexical items that introduce presupposed content may call for different treatment. Presupposition triggers like *too* require a binding with linguistic material, whereas this is not the case for presupposition triggers like definite descriptions:

(14) a. Nico lives in New York. Natalia lives in New York too.
b. It's not the case that the most famous philosopher of antiquity was bald.

It would be semantically anomalous to utter the second sentence of (14a) in an out of the blue context; *too* requires a proposition in antecedent discourse that has a very close and structurally similar content to what is expressed by the clause in which it occurs. On the other hand, a definite description like *the most famous philosopher of antiquity* though it contributes a content that is presupposed (and outside the scope of the sentence wide negation in (14b)) does not require an explicitly mentioned antecedent in discourse to be felicitous. To take another presupposition trigger, Hunter and Asher [8,7] make a good case for treating indexicals as presupposition triggers, but as they make clear the rules for justifying the presupposed content of indexicals differ from those for, say, definite descriptions or factive verbs. In [1], I discuss several kinds of type presupposition justification suited to particular types of type presuppositions, in particular those type presuppositions provided by dual aspect nouns and those provided by predicates that license coercions.

4. The subtyping problem, external and internal semantics

If selectional restrictions involve types and semantic composition involves type checking for semantic well-formedness, we must now integrate the rich system of types demanded by selectional restrictions into our theory of semantic composition. This is not a trivial task, because of the necessity of having a non trivial notion of subtyping. Linguists and computer scientists working on the semantic web entertain lots of subtyping relations—for instance, physical objects are a subtype of objects; animals are a subtype of physical objects, which is a subtype of the type $\textsc{e}$, and so on. Can we simply add such types and this subtyping relation as it stands to the Church/Montague conception of types [5]? The answer is no. For Church/Montague, types are modeled as sets, and subtyping is understood as the subset relation. All of this works fine, as long as we simply look at the subtypes of $\textsc{e}$. But it does not generalize to higher order functional types in any sensible way at all. Functional types $\alpha \Rightarrow \beta$ for Church/Montague are interpreted as sets of functions from the denotation of α to the denotation of β. Consider the type of first order physical properties, $\textsc{p} \Rightarrow \textsc{t}$. It is commonly accepted that *physical* is a subjective adjective, like most other adjectives in the world's languages; a physical threat is a threat. If we recast this observation in type theory, an entity that is of type $\textsc{physical threat}$ should also be a type of, and hence a subtype of, $\textsc{threat}$. *Mutatis mutandis*, first order physical properties are then a subtype of first order properties—$(\textsc{p} \Rightarrow \textsc{t}) \sqsubseteq (\textsc{e} \Rightarrow \textsc{t})$. But this is not so on the Church/Montague set theoretic model of types. In fact the type of first order physical properties and the type of first order properties are disjoint according to Church/Montague. The only subtyping relation that holds in the Church Montague framework for higher order types α is that $\alpha \sqsubseteq \alpha$. The difficulty is to generalize the intuitive subtyping relation over subtypes of $\textsc{e}$ to higher order functional types.

Staying within the set-theoretic paradigm for modeling types appears difficult if we want to accommodate all of our intuitions. Perhaps we could resort to interpreting higher order types via sets of partial functions, but this threatens the consistency of the whole system as the absurd type $\bot$ appears to then have an inhabitant, the empty partial function.

We can also consider a many typed theory TY^n, on analogy with TY^2 of [6], where the types $\textsc{e}$, $\textsc{t}$, and $\textsc{s}$ are disjoint atomic types. There's no subtype structure over these types, so the problem discussed above doesn't arise. On the other hand, we want a sub typing relation for checking semantic well-formedness and we need, as we've just seen, to have the type information encoded within selectional restrictions considered differently from proferred content. Méry [13] investigates

[4] See [3] for a survey.

the idea of taking TY^n as base and then integrating this with Girard's system F. The subtyping relation is defined first by stipulating morphisms between these atomic types and then taking their transitive closure.

While system F is a powerful system in which many things can be represented, it doesn't say anything about *how* a system of types for linguistics should be encoded or what such a system should even be. All the delicate conceptual work remains to be done. Furthermore, in System F, types are still considered rather extensionally and can be modeled as sets. I think that finally a different approach to types are called for. There are good reasons for separating out the denotational content of terms and their type-theoretic meaning in natural language semantics. The standard semantics of terms for natural language semantics is done with denotations, whether they be extensions or intensions. These are externalist meanings and have an important role to play. The semantics of types involves another, internalistic notion of meaning and has a different role to play.

Consider, for example, the set of fictional entities. Let us take seriously the idea that fictional objects are indeed fictional and so they don't exist; they are not part of the domain of interpretation. Further, fictional objects not only don't exist in the actual world; they don't exist in possible worlds either—i.e., other ways in which the world could be. A fictional creature like a hobbit is not at all like the sister that I might have had; the latter exists in a possible world, the former does not. That's what it is to be fictional. So on a view that identifies types with their inhabitants, types corresponding to fictional objects and the absurd type would be the same type, since they have the same extension or the same set of inhabitants, namely the empty set. But types of fictional objects are intuitively distinct from $\bot$.

Whether a term describes a fictional character or not certainly appears to make a difference as to how predications are understood. Within fiction, there is no question of checking or wondering whether the predication actually results in a literal truth. It is even quite controversial among philosophers who have written on fiction whether terms that appear to refer to fictional entities refer in fact to anything at all. On the other hand, fictional talk differs from metaphorical or loose talk; fictional talk is literal—the trees in *The Lord of the Rings* literally speak (see 15) whereas in metaphorical talk (16) the predications aren't to be taken at face value.

(15) Look! the trees are speaking. (*Lord of the Rings*)
(16) These trees are really speaking to me. I'm going to paint the living room green.

To make sense of this difference in predicational behavior, we should distinguish a type of fictional objects. And it should be distinct from the absurd type no matter what the circumstances of the actual world are. This leads me to adopt the thesis that types are neither to be identified with their actual inhabitants (extensions) nor even their possible inhabitants (standard semantic intensions). They are "hyper-intensional."[5]

If types are intensional entities, then, they are not intensions as semantics standardly conceives of them—i.e., as functions from indices (possible worlds or sequences consisting of a world, time, context, and perhaps other elements pertinent to semantic evaluation) to extensions.[6] So given the relatively well-understood analysis of properties in formal semantics and pragmatics as semantic intensions (or as functions from indices to extensions), types cannot be properties either. In addition, properties are typically understood to be mind-independent entities, entities whose existence is not dependent on the existence of minds. Types, however, given their role in guiding predication, are part of the conceptual apparatus necessary for linguistic understanding. They are *mind-dependent representations* of mind-independent properties and individuals. This leads us to the hypothesis that types are concepts, which I take to be mind-dependent entities as well.

I suggest that types are concepts, or at least very similar to them. Concepts, like types, come at different levels and granularities. They form a hierarchy, just as types do. There are concepts of what it is to be a property, what it is to be an individual, what it is to be a physical object, and so on. There are also much more specific concepts: the concept of red, the concept of Ségolène Royale or of Hillary Clinton. Like types, concepts are the internal, mind dependent reflection of mind independent properties and individuals they are concepts of. Concepts (and types) have their own "internal" semantics which has to "track," in the appropriate way, the properties and individuals they are concepts of. It is in virtue of such tracking that a concept is a concept of some object or property. For example, the concept of red tracks the property of being red. A concept RED of the color red is triggered by something at a particular location l in the conceiver's visual field, typically when the conceiver is perceptually aware of something at l that is in fact red in color. Though this tracking is generally reliable, it can occasionally fail. For instance, if there were something wrong with a conceiver's visual apparatus or the circumstances of the perceptual event were very non-standard, the concept RED could be introduced as holding of some object when the object of which the conceiver is perceptually aware is in fact green in color. Another way the tracking could go wrong is that the object is in fact red but the concept fails to be triggered.

Although making this notion of "tracking" precise is a book-length project and not one pertinent to semantics but rather to the philosophy of mind, I can say a few words here. I understand this notion in terms of how the rules for the application of the concept in the conceptual system function. In fact it is these rules that define the concepts and give them their content. These rules look something like natural deduction rules. A concept has certain "introduction rules" and

[5] Reinhard Muskens in [14] also argues for a hyper-intensional construal of types.

[6] The extension (at an index) of an individual constant is an individual; the extension of a 1-place predicate is a set; and an extension of a closed formula is a truth value.

certain inference rules that it licenses. The introduction rules stipulate that an object must satisfy certain conditions for its falling under a certain concept; these conditions may be determined by the sensory system of the organism or by other associated concepts. Talk of satisfying the application conditions of a concept can be replaced by the notion of something's being provably of that type, giving an intuitionist flavor to the interpretation of types.[7] There are more complex combination rules as well that determine how one concept interacts with others. It is these same rules that determine whether an object is of a basic type like CAT or not. Type presuppositions and the rules for type presupposition justification are instances of rules of type and concept combination. This system of rules supplies what computer scientists call an *internal* semantics of a term and these rules define or give the content of concepts and *a fortiori* of types. This internal semantics contrasts with the mind-external, denotational semantics of the terms, which involve real world objects, properties, and so on. The two are connected by the tracking mechanism. Linking concepts and types together helps us understand both: concepts get a rigorous framework from type theory, while types are now linked to the agent's sensory interactions with his environment and as well as the interactions between other concepts/types in the linguistic system.

It is plausible that humans in a given speech community share concepts and a type system. They must do so in order to communicate, to exchange information.[8] The internal semantics conceived as a system of proof or computation rules allows us to make sense of a shared conceptual system. If your concept of red and my concept of red have the same internal semantics, then we can be said to have the same concept of red, and similarly for other concepts. We can prove of two such proof or computational systems whether they are the same or not, using several different criteria. The crudest one is input/output equivalence; roughly two systems are input/output equivalent, just in case they give the same results in the same cases. With respect to the conceptual/type system, this would mean that the same linguistic actions and judgments are observed in the same contexts. Demonstrably, members of the same speech community have systems that are largely input/output equivalent. There is also the criterion of trace equivalence, where for each computation, the same sequence of actions is observed in the two systems. Finally, there is the criterion of bisimulation, according to which for each point in the computation there are the same possible continuations.

Because the rules that make up the internal content of concepts are in general defeasible, concepts cannot determine reference to mind independent entities or properties independently of the context of their application. My notion of a concept thus differs from Frege's notion of a sense.[9] On a standard Fregean view, senses compose together to yield thoughts, which determine on their own the truth conditions of sentences and discourses. I propose a different view. A proposition is the result of the compositional interpretation of logical forms for the words that make up the sentence or discourse. But the compositional interpretation of a logical form results for me, as for most semanticists, in an intension—a function from indices to truth values. Since concepts don't determine extensions, let alone intensions, concepts cannot be the constituents of propositions.[10] Philosophers might take intensions to be simply formal stand-ins for what propositions "really are." But even then, if sentences are typically about mind-independent objects and the properties and relations these objects stand in, then the "real propositions" such sentences express will not contain concepts either—at least not of the sort I have in mind, mind-dependent entities with an internal semantics. In effect, for natural language we need *two* semantics, one to guide the construction of logical forms, the other for evaluation of content and success.

More concretely, consider basic referential expressions: indexicals, proper names, demonstratives. The content of these terms, intuitively, has to do with the individuals they denote, not some proof object or set of rules for defeasibly determining whether a given object is in their denotation. The content of the type associated with *you* consists of rules for determining who the audience is in a particular context. But that's not the contribution of *you* to the content of a clause in which it occurs. Its semantic content in this sense is the audience itself.

Forceful externalist arguments given by Kripke [9], Putnam [17], and others show that our concepts associated with names of individuals and natural kinds do not suffice to determine the extensions or intensions of these expressions. If one looks to the behavior of such terms in modal contexts, there is compelling evidence that their meanings are not in general determined by "what is in the head" of a competent speaker of the language. In this respect too, types resemble concepts; they are tied via the expressions they type to properties and real world entities, but they are not identical to properties or real world entities, nor to sets thereof. They are part of our conceptual apparatus used to guide predication.

To show how the externalist arguments affect types, let's consider a typical Twin Earth scenario, familiar from the externalist literature cited above. Oscar on Earth and his twin "Twin Oscar" on Twin Earth speak syntactically identical languages and are type identical down to their molecular constitution. In keeping with general physicalist principles, they have the same internal make up, the same thoughts, the same conceptual system. In particular their linguistic judgments about semantic well-formedness will be the same. Thus, when Oscar and Twin Oscar each interpret the strings *water is wet* and *water is a tree*, they assign the same syntactic form and the same semantic types to each expression; for the first string they will each construct a coherent logical form using the tools of the theory of predication, while for the second they will not. But whereas they marshal the same type system and conceptual resources when dealing with their languages, the

[7] More on this below.

[8] A proof of this fact is to be had in [10].

[9] Peacocke [15] uses concepts as constituents of thoughts to account for informativeness and Frege style puzzles about the substitution of coreferential terms; he takes concepts to be something like Fregean senses.

[10] In fact, on the standard semantic conception of propositions, propositions don't have "constituents" except in a set theoretic sense—and these would be sets of n-tuples of worlds, other indices, and truth values.

languages of Oscar and Twin Oscar are different—they have a different semantics. On Earth *water* picks out the kind H_2O, or real water, whereas on Twin Earth the string *water* has a different semantics—it denotes a chemical compound distinct from H_2O, which, so the story goes, is XYZ or "twin water." *Water* in English picks out a different substance from *water* in Twin English. (Oscar and Twin Oscar, however, live at a time when chemistry is relatively undeveloped and so are not aware of these differences.) Thus *water* makes dramatically different contributions to truth conditions in sentences of English and Twin English like:

(17) Water is H_2O.

(17) is true (and necessarily true) in English but false (and indeed necessarily false) in Twin English. Such Twin Earth scenarios are well established in the philosophical literature, and intuitions about them are relatively robust. They constitute powerful evidence that as internal reflections of properties and individuals, concepts, and types are not identical with mind-independent properties or individuals nor do they determine them, although they are associated with them through the tracking mechanism.

If concepts, and *a fortiori* types, are not constituents of propositions, they can nevertheless compose together. Making the linguistically relevant types a subset of the set of concepts allows us to use the logical framework of types to explore concepts. Types associated with properties are functions from one type into another; when given an appropriate type as argument, types associated with properties return a new type. We can even compose types or concepts together to give us types associated with propositions or semantic intensions.[11] I shall call the type corresponding to a proposition a thought. If there are as many types as there are distinct word stems in the language, then the hypothesis that types are concepts and compose together to yield thoughts gains in plausibility.

Let us look a bit more closely at the composition of thoughts. Once we distinguish between subtypes of the type E of entities non-extensionally, we must think of the complex types in a more fine-grained way as well. Just as we have different subtypes of the type E, we also have different subtypes of $\text{E} \Rightarrow \text{T}$, or even T itself, the type of propositions. For instance, the concept or type RED associated with the nominal modifier *red* combines with other concepts to form more complex concepts like RED PEN, RED APPLE, and so forth. This is their "proffered" internal content that they contribute to the thoughts of which they are constituents. The nominal modifier *red* has a type that is a subtype of $(\text{E} \Rightarrow \text{T}) \Rightarrow (\text{E} \Rightarrow \text{T})$. *Apple* itself has a type that is a subtype of $\text{E} \Rightarrow \text{T}$. Later we will see how these "proffered," as opposed to presupposed types affect the calculation of logical form.

It is important to separate these proffered types of terms from the type presuppositions of these terms. Both *red* and *apple* have type presuppositions on their arguments in logical form. For *red* its argument must be a physical object; for *apple* its λ bound variable must be an offspring of a plant. This allows such terms to enter into semantically well formed predications that are necessarily false (on an externalist semantics) like

(18) Apples are vegetables.
(19) Those are apples. (said by someone pointing to pears)

but prevents the following from being well-formed predications when the words are understood literally:

(20) #Apples are rocks.
(21) #Those are apples. (said by someone pointing to cars)

Here's an approximation of the logical forms I shall assign to *red* and *apple*, with their type presuppositions expressed as restrictions on the λ bound arguments: I have assigned types to all the bound first order variables in them by writing, for instance, $x\text{: P}$, which means that x has the type PHYSICAL OBJECT[12]:

- $\lambda P\text{: E} \Rightarrow \text{T}\ \lambda x\text{: P}\ (\text{red}(x) \wedge P(x))$.
- $\lambda y\text{: PLANT-OFFSPRING}\ \text{apple}(y)$.

This means that *apple* is defined when it applies to fruits other than apples to yield a truth value (it yields the truth value "false" in such cases).[13]

The fine-grained type APPLE is a function in intension that can be used to evaluate whether an object is an apple or not. APPLE is a primitive type for the type system, but I also suppose that there is a function EVAL which when applied to a type and an object, or rather an individual concept of an individual object, returns a "proof" that the object is an apple,

[11] Composition allows us to talk of concepts as constituents of thoughts, though this talk should be interpreted with care. It is true that concepts compose together to form thoughts, but that does not necessarily mean that the finished product will actually contain those concepts. Nevertheless, because I shall identify thought contents at least in part with the derivation of the thought from its constituents through composition, that's pretty close to constituency.

[12] An anonymous reviewer of this paper suggested investigating the typing conventions of the calculus of constructions, which is an interesting idea to pursue in the future.

[13] This is illustrative only. Nothing hangs on the particular choice of the type presupposition for *apple*, only that it must have one.

if that object meets the conditions of application, and returns a "refutation" that the object is an apple, if it fails to meet the conditions of application. It may also fail to give a value, say for borderline or unfamiliar cases.[14] It is this fine-grained type that is the internal proffered content of the term *apple* and that enters into the composition of thoughts. The same is true of all primitive types. The basic, primitive types like CAT, DOG, ANIMAL are analogous to the defined types like INTEGER in mathematical applications of the λ calculus. The difference is that concepts and *a fortiori* types don't seem to have easy definitions in the ordinary sense, although there are computations that humans have mastered that enables them to use concepts/types correctly. When I write where v is a variable, v: APPLE, I mean that the object denoted by v relative to an assignment meets the application conditions that are constitutive of the type APPLE; thinking of these application conditions as a proof or computation, this means that when the computation takes the value of v as input, it terminates and yields the value *true*. The notion of "object" within the internal semantics should not be confused with "real objects", as the former are individual concepts.[15]

What about the fine-grained type RED? RED is a functor that when applied to a subtype of P returns another subtype of P—exactly which subtype of P depends on the fine-grained type of its argument. The type RED thus has an argument place in it for a physical type. When the functor is applied to APPLE, we get the physical subtype:

- RED(APPLE).

Once again, we can evaluate this concept relative to some object. We now define the fine-grained type and internal semantics of the word *red* vis-à-vis the polymorphic type RED. RED is a polymorphic type because the particular type of RED depends on the type of the modifier's argument. RED presupposes that its argument be a subtype of P. If the modifier combines with a noun whose proffered type is α and α is a subtype of P, then RED(α), the application of RED to α, is the proferred type of the modified noun. We may go further and require RED to be a *dependent* type, as in

(22) John hates everything red—red meat, red apples, red shirts, and so on.

The exact type given by RED will in this case depend on the value of the variable. While most of the time adjectives can be understood as having polymorphic types, we seem to need dependent types to handle quantificational examples like (22). If, as in system F, we allow ourselves abstraction over types at least restricted to being subtypes of a given primitive type, then we can express the type expressed by *red* as

- $\Lambda\alpha \sqsubseteq$ P RED(α).

In virtue of the meaning of RED, which is intersective with respect to the internal semantics (of types) as well as to the external semantics (of intensions), we have as a postulate:

- RED(α) $\sqsubseteq \alpha \sqsubseteq$ P.

The fine-grained types of verbs are similar to RED in that they are functors taking a sequence of types (the sequence represents their arguments) and returning a subtype that involves the contribution of the verb applied to those arguments. It is natural to think of verbs as the "glue" that binds the other types in a thought together. This means we must introduce more complex types for verbs than is usual in standard formal semantics. It is the DP rather than its head noun that furnishes the type of the argument to the verb, since determiners are often responsible in a language like English for saying whether the argument of a verb is count or mass, or a plural that is collectively interpretable or only has a distributive interpretation. The type of a verb takes DP types (or the types associated with the DP's values) as arguments to give us a proposition. Thus, intransitive verbs will be functors from one DP type to propositions, while transitive verbs will be functors from two DP types. Differences in verb alternations can be represented by the differences in the output that a particular instance of the polymorphic or dependent type yields.

To figure out what a DP type is, we must turn our attention to determiners. Their conceptual role is to state relations between types in terms of the individual concepts that satisfy the application rules for these types. Each determiner with a distinct logical function will give rise to a distinct determiner concept. For instance, the determiner *every* has a fine-grained type EVERY that when applied to a subtype of COUNT $\sqsubseteq$ E returns a functional type that, given type that is a function from DP types to PROP, returns a thought, or a subtype of the type of propositions, PROP. The thought expressed by applying EVERY first to α and then to β is the thought that every individual concept that satisfies the application rules of α also satisfies the application rules of β. The internal semantics of EVERY is a type theoretic version of a [2] generalized quantifier. An intransitive verb or VP type takes a DP type as an argument and provides it with a subtype of E $\Rightarrow$ PROP to yield a thought. For example, the sentence in (23a) has the corresponding thought described by the fine-grained type in (23b):

[14] This differs from what advocates of Type Theory like Martin-Löf [12] and Ranta [18] take the denotation of a λ term to be; in type theory the denotation of a term is the collection of "positive" proofs. This corresponds closely to the notion of a classical extension.

[15] Cardelli [4] takes an object is essentially a collection of functions or attributes with values, but there are other construals of individual concepts that are compatible with this approach.

(23) a. Every dog barks.
b. BARK(EVERY(DOG)).
c. $\forall x(\text{dog}(x) \rightarrow \text{bark}(x))$.

From (23b) we determine the type of *bark*. It is the function from DP types α to the fine-grained type BARK(α), which is a subtype of the type of propositions. This generalizes straightforwardly to other grammatical categories.[16]

Let us examine whether we need abstraction over types in the way that System F proposes. I think we do not, because finally our goal is not to have logical forms for the internal type semantics. We can establish the internal thought corresponding to a sentence and understand the meanings of expressions simply as functions of the right sort. What we need, however, is to be able to place type requirements on terms of the traditional logical forms for clauses and discourses. And that we can do without resorting to abstraction over types. If the logical form for expressing the internal semantics of RED requires abstraction over types, the ordinary logical form with presupposed types added for type checking clearly does not:

- $\lambda P\colon \text{P} \Rightarrow \text{PROP}\ \lambda x\colon \text{P}\ (P(x) \wedge \text{RED}(x))$.

So to conclude, types are concepts not identified with sets of their inhabitants but rather something like proof objects with rules of application.[17] Types furnish the internal semantics for logical forms and provide constraints on semantic well-formedness. They can even furnish conditions for *a priori* truth, a form of analyticity (an analytic proposition is one whose truth or falsity can be determined from linguistic conventions and logic alone), as argued in [1], though the notion of analyticity here is perforce somewhat different than that usually found in philosophy. The reason for this has to do with the failure of concepts to determine extensions; so here a sentence's being *analytic* means if the types involved in the thought it expresses pick out under EVAL the actual extensions of the terms, then the sentence is true in virtue of the thought it expresses.

My construal of types in natural language has an analogue to the way types are construed in constructive approaches to mathematics [12,11]. With mathematical objects, it is clear what are their rules of application; something has the type INTEGER iff it can be shown to be equal to the result of applying the successor operation to 0 a finite number of times; a predicate has the type of a functor from integers to lists iff it provides a procedure for turning anything of type integer into a list, or two objects are bisimulation equivalent just in case there is a proof of a bisimulation between them. Also, it is much less clear in mathematics that there is a failure of mathematical concepts or types to determine extensions and hence truth (though some have argued that even in mathematics there are truths that cannot have any effective proof). For natural language types/concepts it is much less clear what are the rules of application, though linguistics and intuitions provide us with ideas about what concepts are subtypes of others. It is perhaps just because of this unclarity that the failure of concepts to determine extensions receives strong support from thought experiments like the one above.

Category theory is an elegant framework in which mathematical and natural language types can be represented. Category theory is ideal for concepts, because it allows us to characterizing them without details of internal structure. A Cartesian Closed Category furnishes the basics, providing a model of $\Rightarrow$, $\vee$ and thus functional and disjunctive types, the objects of the CCC being given by the basic types. Instances of a type α are modeled by arrows, $1 \rightarrow \alpha$. A topos allows us to model the quantifiers as well as the positive Boolean connectives. It also contains the sub object classifier Ω, which also serves as the type of truth values. But now how should we think of complex concepts, in particular PROP? In embedding Montague Grammar within category theory, Lambek proposed to identify sentence meanings with $1 \rightarrow \Omega$. This yields only a purely extensional theory, with just two propositions! This is fine if one equates propositions with truth values, but we don't want to equate concepts or thoughts with truth values. So with Pollard [16], we distinguish a Pre Boolean Algebra object PROP distinct from Ω. PROP/$\equiv$ is a Boolean Algebra, where $\equiv$ is the equivalence relation induced on PROP by the usual Boolean identities. PROP is ensured to be non-empty by inductively constructing its inhabitants from a basic set of types using the types used for natural language expressions together with their type presuppositions.

- if $a_1, \ldots, a_n$ are types meeting type presuppositions of the n-arguments of a predicate concept τ, the $\tau(a_1, \ldots, a_n)$ is in PROP.
- PROP is closed under concepts for Boolean operations, and other operators and quantifiers (over types) should we wish.

To this picture I add the functor EVAL: PROP $\rightarrow \Omega$, which gives the extension of the proposition. While all of our primitive types produce elements of PROP, we can recreate the usual extensional types by taking EVAL(PROP). Because the elements of PROP have an internal structure to them of the sort indicated in (23b), a map from EVAL to can be given that captures the usual truth clauses for quantifiers and connectives

[16] Notice that quantifier scope is not necessarily resolved in the type, whereas it perforce is in the standard logical form; this is of course a choice—I could have done things otherwise. But since types do not determine extensions, the underdetermination at the level of thoughts is not a drawback. Indeed it may be an advantage, if we wish to define a "natural logic" over thoughts that is weaker than the logic validated by the classical semantics of standard logical forms.

[17] As an anonymous reviewer has pointed out, there is a problem if there is no proof that a given object is of a certain type. Ludics may offer a way to address this problem. I do not address this difficulty here, however.

Let's now go back to subtyping. The notion of subtyping needs a bit of work in this framework. The problem has to do with an "overloading" of the notion of subtype: it is to be linked to deduction, but it is also supposed to play its traditional role in the type hierarchy—he set of types ordered under a subtyping relation $\sqsubseteq$, which is usually assumed to be at least transitive and reflexive and often also antisymmetric, something which I also assume). The category theoretic model gives us a notion of deduction through morphisms:

- a simple type A gives us instructions for finding morphisms $1 \to A$.
- $\alpha \sqsubseteq \beta$: a proof that something is α furnishes a proof that that thing is also β.
- $\alpha \Rightarrow \beta$: a proof given something x of type α yields a proof that x is of type β.

But we don't just want any deduction to provide us with a subtype. For instance, consider the type PROP $\Rightarrow$ PROP. Its translation in propositional logic is a theorem as is PROP $\Rightarrow$ (PROP $\Rightarrow$ PROP). Simply using the full deductive apparatus of a topos would mean that the type of say the truth functional connective *and* is the same as the type of a one place operator like negation or modality. This would predict that *John loves Mary and* is a sentence expressing a well formed semantic proposition, when it does no such thing.

So I restrict the set of arrows relevant to subtyping to those just generated over a given type hierarchy ST for the simple subtypes S of E. That is $ST = \langle S, \sqsubseteq \rangle$, where $\sqsubset \subseteq S^2$ is a partial order over S. From the perspective of deduction, simple types are just propositional variables. The $\to$ below signifies a conditional operator, whereas $\Rightarrow$, as usual, stands for the corresponding functional type constructor. I further restrict derivations $\vdash$ to only primitive introduction and elimination rules on the premises given (the use of theorems to derive conclusions is not permitted).

- $\Delta = \{\alpha \to \beta \colon \alpha \sqsubseteq_{ST} \beta\}$.
- $$\frac{\Delta,\ \alpha \vdash \beta \text{ and } \Delta \nvdash \beta}{\alpha \vdash_{\Delta} \beta}.$$

Definition 1. Subtyping: $\alpha \sqsubseteq \beta$ iff $\alpha \vdash_{\Delta} \beta$.

This definition gives us the well known rule of contra variance for functional types:

Fact 1. Subtyping for functional types*:*

$$\frac{\alpha \sqsubseteq \alpha' \quad \beta \sqsubseteq \beta'}{(\alpha' \Rightarrow \beta) \sqsubseteq (\alpha \Rightarrow \beta')}.$$

This is still not what we want for the relation between first order properties and physical properties. But there is an easy fix. Let us define first order properties with restricted quantification over types:

- $\Sigma x \subseteq \text{E}\ (x \Rightarrow \text{T})$.

and physical first order properties are defined as

- $\Sigma x \subseteq \text{P}\ (x \Rightarrow \text{T})$.

where Σ is an existential quantifier over types but restricted by a super type; if we suppose that the number of subtypes of any given type is finite, this reduces to a disjunction over types. Given that P $\sqsubseteq$ E physical properties are first order properties, which is what we intuitively desired.

5. Type systems for lexical semantics

Within a topos, I have singled out the wherewithal to construct an adequate system of types for lexical semantics. To handle basic logical form construction and basic homonymous cases of lexical ambiguity, [1] shows how to use the following:

- **Simple or Primitive Types:** E, T, PHYSICAL OBJECT, etc., with a linguistically motivated subtyping relation $\sqsubseteq$ defined over these types.
- **Disjunctive Types:** If σ, τ, and ρ are types, $\sigma \sqsubseteq \rho$ and $\tau \sqsubseteq \rho$, then $(\sigma \vee \tau)$ is a type.
- **Functional Types:** If σ and τ are types, then so is $(\sigma \Rightarrow \tau)$.
- **Quantificational Types:** If σ is a simple type, and τ is any expression denoting a type and x is a variable ranging over types, then $\exists x \sqsubseteq \sigma\ \tau$ is a type.

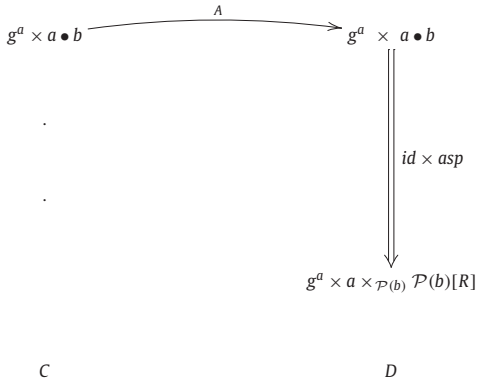

Fig. 1. The natural transformation *A* and its associated arrow *aspect*.

This is a rather small system sitting inside a topos. It is clearly a small part of System F. But it is also different from system F in that it has a very different interpretation and orientation, or from classical Martin Lof proof theory. Given the two level semantics, the link between type semantics and denotations of terms is also different. In modern type theories, of course, the internal and external meanings in my senses are identified but in non homogeneous ways. For example singular terms refer to objects, while propositions refer to proofs. In the system that I propose there are two semantics, but they are homogeneous: the external semantics is concerned with denotations—where individual terms refer to individuals (or constant functions from indices of evaluation to individuals), and predicates to appropriate extensions (or functions from indices to appropriate extensions); the internal semantics is concerned with concepts—individual terms are interpreted as individual concepts and predicates as functions from individual concepts to propositions, construed as proofs. Unlike Church's original system, I make the link between internal and external semantics much more indirect and with a much more developed conception of concepts or types. In effect I have changed the way of thinking what concepts are supposed to do: for me they guide the construction of logical form and provide certain analytic inferences; for Church concepts were supposed to do all the work that Fregean senses were supposed to in semantics, a task for which they are ill suited.

There are also some things that an adequate type theory needs that a topos doesn't have "on the shelf"—in particular type constructors that license natural transformations of the category in the presence of other types; we need these types to deal with phenomena involving dual aspect nouns like *book* and the vast array of coercions that exist in natural languages. Fig. 1 presents a schematic picture of such a transformation. For example, for dual aspect nouns, I require a primitive type constructor $\bullet$ which, in the presence of an appropriate particular predicational environment, licenses a transformation of $\alpha \bullet \beta$ into a particular pullback. That is, when a term of type $\alpha \bullet \beta$ is an argument of a predicate that has a selectional restriction for a term of type α, we get a particular pullback structure. Predicates that license coercions introduce a similar transformation. By exploiting the morphisms present in a pull back, we can pick the appropriate aspect of *book* to use either with predicates that presuppose an argument of physical type P or with predicates demanding an argument of informational object type I. The operation $\bullet$ is a type constructor licensing a pull back in particular predicational environments.

Let's consider a $\bullet$ type that concerns us, the type $\text{P} \bullet \text{I}$. The natural relations between informational and physical aspects that restrict the pull back are exemplification and its converse, instantiation. These natural relations are not necessarily functions, but become functional once we lift them to power objects $\mathcal{P}(\text{P})$ and $\mathcal{P}(\text{I})$; i.e., *instantiates*: $\text{P} \rightarrow \mathcal{P}(\text{I})$ and *exemplifies*: $\text{I} \Rightarrow \mathcal{P}(\text{P})$. The pull backs used to model $\bullet$ types are special in two ways. First, they are *degenerate*, in that the "c" objects mapped to by the morphisms *instantiates* and *exemplifies* are always isomorphic to the power object of P or of I. Secondly, my pull backs are restricted via the natural relations that are used to define the $\bullet$ type. Not just any function from the power domain of one object into another object will provide a pullback associated with a $\bullet$ type. Only those functions that encode relations that are part of our common sense metaphysics, like instantiation and exemplification will do. The view is that $\mathcal{P}(\text{P}) \times_{\mathcal{P}(\text{P})} \text{I}[\textit{instantiates}]$ models $\text{P} \bullet \text{I}$ in its informational guise, and $\text{P} \times_{\mathcal{P}(\text{I})} \mathcal{P}(\text{I})[\textit{exemplifies}]$ models the type in its physical guise. More concretely, if a book is subject to a predicational environment where an I aspect is needed, the transformation *A* yields a pull back individuated relative to information content. Each "i-book" will map onto the collection of its physical aspects in $\mathcal{P}(\text{P})$. The pull back $\mathcal{P}(\text{P}) \times_{\mathcal{P}(\text{P})} \text{I}[Ex]$, where *Ex* stands for the exemplification relation, counts as the same all those perhaps physically distinct books that have the same information content—thus corresponding to books informationally construed, while $\text{P} \times_{\mathcal{P}(\text{I})} \mathcal{P}(\text{I})[In]$, where *In* stands for the instantiation relation, counts as the same all those books that have perhaps differing information contents but occur in the same physical volume.[18]

Let's consider first the "informational" pullback of $\text{P} \bullet \text{I}$ in Fig. 2, with *Ex* standing for the exemplification relation. Given the *id* map from $\mathcal{P}(\text{P})$ to itself, the characteristics of pull backs force $\pi_1 = \pi_2 \circ \textit{exemplifies}$ and π_2 to be a monomorphism; this means that the $\bullet$ type acts like an I type.

[18] For more discussion of the linguistic data supporting this, see [1].

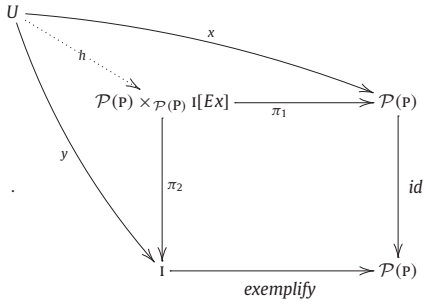

Fig. 2. The "I" pullback.

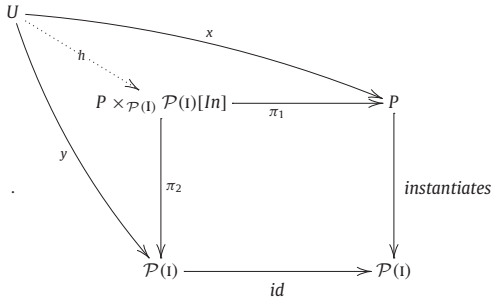

Fig. 3. The "P" pullback.

It is not just the typing of the term that is affected by this transformation. Recall that the types exploited by selectional restrictions and type presuppositions encode individuation conditions. These individuation conditions affect denotations insofar as the conditions for individuating objects will determine the domain of quantification associated with the type. In particular, this means that the number of books when individuated informationally may well be distinct from the number of books individuated physically. Under the transformation in Fig. 2, inhabitants of P • I are counted in the same way as I objects are.[19]

When books are required to act like physical objects because of the demands of the predicational environment, we have a slightly different pullback (Fig. 3). Under this morphism, the individuation conditions for physical aspects of books determine the number of inhabitants of the type (by determining the number of the elements of the image of $\mathcal{P}(P)$ under $\pi_1 \circ t$). The "physical" pullback of P • I restricted by the instantiation relation *In* in Fig. 3 forces π_1 to be a monomorphism. When both aspects are exploited in the predicational environment as in copredication, we have two pullback squares off a common • type, and the counting principles for objects may well be different in the two squares. Thus, counting principles for inhabitants of • types will shift with the local predicational environment, something which [1] shows to be required by the linguistic data. As far as I am aware, no other type system allows such counting variations.

For coercions like (24), the story is similar. Certain predicates have a type presupposition that allows a natural transformation of their given argument into another type. This type presupposition is modeled via another type constructor that licenses the introduction of another term with the coerced type within certain predicational environments. This product construction does not affect counting principles, unlike the cases of exploiting the type information in dual aspect nouns to justify type presuppositions. Furthermore, this transformation crucially does *not* change either the argument's type or the predicate's type, which allows us to handle anaphoric references as in (25), and various ellipses as in (26)

(24) John enjoyed the book (the coercion involves an event—this sentence means something like *John enjoyed doing something with the book*).
(25) John started (doing something presumably reading) *War and Peace*, you know the book Tolstoy finished (writing) only in his old age. But it (John's reading the book) won't last, the book's too long for John to keep at it.
(26) a. I'm parked out back. Mary's car is too. (VP ellipsis precludes shifting the meaning of *parked out back*.)
b. #I'm parked out back and am an old Volvo (if we shifted the meaning of the argument *I* then this copredication should be good when it is not).

[19] As usual, each inhabitant in the type is understood as an arrow from the terminal object to the type.

Once we have introduced these new type constructors and their transformations into the type system, the question of whether the contents of internal semantics and external semantics are preserved under normalization and the application of various morphisms licensed by the type constructors. Normalization and the application of morphisms associated with the complex type constructors preserves the internal semantics; a term of • type t remains of that type; the application of the morphisms are just used to check that the typing is compatible with the predicational environment. The interaction of morphism applications with the external semantics is more delicate, but there too the semantics is preserved, because the selection of a particular aspect introduces a new term for the aspect that is related to t. The term t retains its original type and its original denotation even after the application of morphisms; what the morphisms do is introduce other terms and other denotations into the semantics. A similar operation occurs with the morphisms associated with coercions.[20]

6. Conclusions

I have discussed here some possible categorial foundations of lexical semantics. The picture I propose exploits categorial structures for an internal semantics of natural language which uses types, understood as proof objects. Category theory is a nice tool for modeling the structure of these objects and the relations between them, since we cannot at present say in detail what the proof rules associated with individual words in a language are. The constructions I make use of for defining • type constructors require certain aspects of the categorial structure of a topos, the ability to define at least restricted quantification over types (it may be countably infinite when nominalizations are considered), the closure under pull backs with certain relations, exponent and product.

The work sketched here that develops some aspects of the approach in [1] provides several interesting directions for further research. The first is to investigate in more detail what the conceptual content of types are. Another direction, which is complementary to the first, is to develop links between a symbolic conception of types and more empirical conceptions that can be gleaned from distributional methods, which are currently a source of considerable excitement in computational linguistics, and in particular in computational lexical semantics.

References

[1] N. Asher, Lexical Meaning in Context – A Web of Words, Cambridge University Press, 2011.
[2] J. Barwise, R. Cooper, Generalized quantifiers in natural language, Linguist. Philos. 4 (1) (1981) 159–219.
[3] D. Beaver, Presupposition and Assertion in Dynamic Semantics, CSLI Publications, Stanford, 2001.
[4] L. Cardelli, A semantics for multiple inheritance, Inf. Comput. 76 (1988) 138–164.
[5] A. Church, A Calculi of Lambda-Conversion, Princeton University Press, Princeton, NJ, 1941.
[6] D. Gallin, Intensional and Higher-Order Modal Logic with Applications to Montague Semantics, North-Holland, 1975.
[7] J. Hunter, Presuppositional indexicals, PhD thesis, University of Texas at Austin, Austin, TX, 2010.
[8] J. Hunter, N. Asher, A presuppositional account of indexicals, in: P. Dekker, M. Francke (Eds.), Proceedings of Fifteenth Amsterdam Colloquium, ILLC, Department of Philosophy, University of Amsterdam, 2005, pp. 119–124.
[9] S.A. Kripke, Naming and Necessity, Harvard University Press, Cambridge, MA, 1980.
[10] D. Lewis, Convention, Cambridge University Press, 1969.
[11] Z. Luo, Computation and Reasoning: A Type Theory for Computer Science, Oxford University Press, Oxford, UK, 1994.
[12] P. Martin-Löf, Intuitionistic Type Theory, Bibliopolis, Naples, Italy, 1980.
[13] B. Méry, Modélisation de la sémantique lexicale dans le cadre de la théorie des types, PhD thesis, Université de Bordeaux 1, 2011.
[14] R. Muskens, Intensional models for the theory of types, J. Symb. Log. 72 (1) (2007) 98–118.
[15] C. Peacocke, A Study of Concepts, MIT Press, 1992.
[16] C. Pollard, Are (linguists') propositions (topos) propositions?, in: Proceedings of the 6th International Conference on Logical Aspects of Computational Linguistics, Université de Montpellier, Montpellier, France, 2011, pp. 205–218.
[17] H. Putnam, The meaning of "meaning", in: K. Gunderson (Ed.), Language, Mind and Knowledge, in: Minnesota Studies in the Philosophy of Science, vol. 7, University of Minnesota Press, Minneapolis, MN, 1975, pp. 131–193.
[18] A. Ranta, Grammatical framework: A type-theoretical grammar formalism, J. Funct. Program. 14 (2) (2004) 145–189.
[19] G. Ryle, The Concept of Mind, University of Chicago Press, 1949.

[20] For more discussion see [1].