# So what's all this structure good for? Some uses of record types in TTR

**Robin Cooper**
Centre for Linguistic Theory and Studies in Probability
Department of Philosophy, Linguistics and Theory of Science
University of Gothenburg
cooper@ling.gu.se

## Abstract

TTR, a type theory with records, makes use of structured semantic objects such as record types. In this paper we will first explore in what sense this represents an increase in structure over what we normally find in a classical Montague style semantics. We will then mention some of the various uses of record types relating them to $\Sigma$-types in type theory, discourse representation structures, frames, feature structures and various kinds of mental states.

Finally, we will consider how the introduction of such structured semantic objects compares with the kind of proof theoretic approaches common in type theoretical approaches to linguistic semantics on the one hand and on the other hand linguistic theories which introduce a level of logical form or discourse representation structure as a mediation between natural language syntax and model theory. I will try to argue that, in general terms, all these approaches introduce similar notions of structure but that there are advantages to introducing the structure in terms of semantic objects.

## 1 Structured objects in TTR

In this paper we will address and reflect on the use of structured objects in TTR, a Type Theory with Records (Cooper, 2012; Cooper and Ginzburg, 2015; Cooper, in prep). TTR is inspired by Martin-Löf type theory (Martin-Löf, 1984; Nordström et al., 1990) and is a rich type theory in the sense that it does not contain just basic ontological types as in simple type theory such as Montague's $e$ ("entities") and $t$ ("truth-values") and function types defined on these but types of objects like *Tree* and events like *boy-hugs-dog*. Central to such a type theory is the notion of *judging* an object or situation/event, $a$, to be of a type, $T$. In symbols this is expressed as (1).

(1) $a : T$

Types in TTR can be structured objects. One example of this is *ptypes*, types where a predicate is used to construct a type from appropriate arguments to the predicate. Suppose that $b$ is some particular boy and that $d$ is some particular dog. We use (2) to represent the type of situations in which $b$ hugs $d$.

(2) hug($b$,$d$)

The notation in (2) is thought of as representing a *labelled set* (the graph of a function whose domain is a designated set of labels and whose range is $\{\text{hug}, b, d\}$). Thus we encode it as the set of ordered pairs in (3).

(3) $\{\langle \text{pred}, \text{hug}\rangle, \langle \text{arg1}, b\rangle, \langle \text{arg2}, d\rangle\}$

Note that the labelled set in (3) is NOT the set of witnesses of the type. Thus the structure of the type is not given by the set of its witnesses and the type is not identified by the set of its witnesses. If $s$ is a witness for the type 'hug($b$,$d$)', that means that $s$ stands in the *of-type relation* to the set (3).

This gives types the ontological status of mathematical objects (such as sets) whatever you think that is. One view is that mathematical objects are part of the basic furniture of the world. Another view is that they are mental constructs imposed on physical reality. We could also place them in Frege's (1918/1919) third realm. Yet another view, akin to the relational view of meaning of Barwise and Perry (1983) and more recently the relational interpretation of quantum theory (Rovelli, 2021) is that they are inherent in the relation between an observer and the world. Whatever view we take, they should be no more (and probably no less) mysterious than sets. It would probably be a mistake to think of individuals or events as being less mysterious. The boy and the dog in our example do not correspond to any unique collection of physical particles since the physical reality which we identify



*Proceedings of the ESSLLI 2021 Workshop on Computing Semantics with Types, Frames and Related Structures*, pages 3–11
Utrecht, The Netherlands (online), 26–27 July, 2021. 

as these individuals is constantly changing. At best, we can say that these individuals represent strings of events that cohere in a certain way. But what kind of coherence of events counts as identifying an individual seems to depend on the nature of the observer (*cf.* the discussion of schemes of individuation by Barwise, 1989, Chapters 10 and 11 and, for a recent view on how we construct reality from the neuroscience perspective, Seth, 2021).

The type in (2) is the type of situations where a particular boy, $b$, hugs a particular dog, $d$. We use *record types* to represent a more general record type, *boy-hugs-dog*, the type of situations were some boy hugs some dog. Such a record type is given in (4).

(4)
$$\begin{bmatrix} \text{x} & : & \textit{Ind} \\ \text{c}_{\text{boy}} & : & \text{boy(x)} \\ \text{y} & : & \textit{Ind} \\ \text{c}_{\text{dog}} & : & \text{dog(y)} \\ \text{e} & : & \text{hug(x,y)} \end{bmatrix}$$

A record type is a finite set of fields, each consisting of a label and a type (or a dependent type together with a sequence of paths in the type, as we will explain below). The witnesses of record types are *records*. A record of the type (4) could be of the form (5)

(5)
$$\begin{bmatrix} \text{x} & = & \text{sam} \\ \text{c}_{\text{boy}} & = & s_1 \\ \text{y} & = & \text{fido} \\ \text{c}_{\text{dog}} & = & s_2 \\ \text{e} & = & s_3 \\ \dots & & \end{bmatrix}$$

where the conditions in (6) hold.

(6) sam : *Ind*
$s_1$ : boy(sam)
fido : *Ind*
$s_2$ : dog(fido)
$s_3$ : hug(sam, fido)

Records are also finite sets of fields. For a record, $r$, to be a witness for a record type, $T$, for each of the fields in $T$, there must be a field in $r$ with the same label where the value in the record field is of the type in the field of the record type. Note that this allows for there to be more fields in the record with labels not occurring in the type. (This is represented by the '...' in (5).)

In TTR, (4) is a convenient notation for (7) where the dependent fields are spelt out as containing a pair consisting of a dependent type (a function which returns a type) and a sequence of paths within the type (the paths where arguments to the dependent type are to be found in the record being tested against the type).

(7)
$$\begin{bmatrix} \text{x:}\textit{Ind} \\ \text{c}_{\text{boy}}\text{:}\langle \lambda v\text{:}\textit{Ind} \,.\, \text{boy}(v), \langle \text{x} \rangle \rangle \\ \text{y:}\textit{Ind} \\ \text{c}_{\text{dog}}\text{:}\langle \lambda v\text{:}\textit{Ind} \,.\, \text{dog}(v), \langle \text{x} \rangle \rangle \\ \text{e:}\langle \lambda v_1\text{:}\textit{Ind} \,.\, \lambda v_2\text{:}\textit{Ind} \,.\, \text{hug}(v_1, v_2), \langle \text{x}, \text{y} \rangle \rangle \end{bmatrix}$$

Record types in TTR are labelled sets like ptypes, though using a different stock of labels to ptypes. This means that a record type whose graphical display is of the form (8a) is actually the set of ordered pairs (8b).[1]

(8) a.
$$\begin{bmatrix} \ell_1 & : & T_1 \\ \ell_2 & : & T_2 \\ \dots & & \\ \ell_n & : & T_n \end{bmatrix}$$

b. $\{\langle \ell_1, T_1 \rangle, \langle \ell_2, T_2 \rangle, \dots, \langle \ell_n, T_n \rangle\}$

Record types are, then, another kind of structured object in TTR.

Types play an important role in type theories not only as types of objects or situations but also in that they can be used to model propositions. See Wadler (2015) for a discussion of the history of this idea which has a number of sources. The "propositions as types" dictum is of central importance in Martin-Löf type theories. A type, $T$, as a proposition is said to be "true" just in case there is something, $a$, such that $a : T$, that is, $T$ has a witness. As propositions associated with linguistic interpretation are normally taken to be record types in TTR, it follows that we treat propositions as structured objects.

An unconsidered first reaction to this fact is that this makes propositions in TTR very different from the kind of unstructured propositions that one finds in Montague semantics. However, this claim is not true given that we have based our claim of structure on the fact that record types are sets of ordered pairs. For Montague, a proposition is a function from world-time pairs to truth values. That is, a proposition is a set of ordered pairs since functions are modelled as sets of ordered pairs. For us, a proposition can be a record type and, as we have seen, a record type is a set of ordered pairs. So

[1] This is actually a slight simplification. See the discussion of flavours in labelled sets in Cooper (in prep).

what is it that makes us think that TTR's record types are structured objects whereas Montague's propositions are not?

TTR's record types differ from Montague's propositions in two main respects. The first is size. For Montague, the functions modelling propositions are infinite, defined on an uncountable domain, the set of world-time pairs. Record types, on the other hand, are finite functions defined on a (normally small) finite set of labels. The other respect in which they differ is content. For Montague, the proposition *a boy hugs a dog* contains nothing corresponding to *boy*, *dog* or *hugs*. These are used, of course, in defining what function from possible worlds and times to truth values is the proposition but you cannot look at the resulting function and determine what was used to define it. Record types, on the other hand, contain all the elements that were used to build them up in their various fields, so, in a rather trivial sense, you *can* look at the record type and determine what was used to define it. For Montague, then, the process of compositional interpretation involves loss of information, a kind of "catastrophic forgetting" to borrow a term for a rather different (but nevertheless not entirely unrelated) problem from the literature on neural networks. It is precisely this difference which leads us to think of Montague's propositions as unstructured and record types as structured.

## 2 Some uses of structured types

In this section we will briefly review some of the uses to which structured types have been put in TTR.

### 2.1 Types as models of propositions

On p. 2, we introduced the "propositions as types" dictum and explained its importance in type theoretic approaches to semantics. One opportunity this offers, depending on how you define your type theory, is a direct way of dealing with what is often referred to as *hyperintensionality* in the possible worlds approach to natural language semantics. In TTR we allow distinct types to have the same witnesses. Consider the ptypes represented in (9).

(9) a. buy(kim,syntactic_structures,sam)
b. sell(sam,syntactic_structures,kim)

(9a) represents the type of situations where Kim buys Chomsky's 1957 book *Syntactic Structures* from Sam and (9b) represents the type of situations where Sam sells *Syntactic Structures* to Kim. TTR allows us to characterize type systems in which (10) holds.

(10) $s : \text{buy}(a, b, c) \leftrightarrow s : \text{sell}(c, b, a)$

(10) is a restriction on type systems in the same way in which "meaning postulates" in Montague's semantics are restrictions on the possible worlds to which we should direct our attention. In the type theory version, however, we have two distinct types (which can be used as propositions) which have exactly the same witnesses.

This provides us with a good reason to think of propositions as types rather than as sets of possible worlds (or in Montague's version sets of world time pairs). In a possible worlds theory the expression 'buy($a$,$b$,$c$)' intuitively represents the set of possible worlds in which $a$ buys $b$ from $c$. But we have no independent way of characterizing which worlds those are apart from saying that they are the worlds in which 'buy($a$,$b$,$c$)' is true. We cannot say what it is that the worlds have in common which makes them worlds in which the expression is true. Types, on the other hand, provide a theory of what situations (or possible worlds) might have in common and then expressions are related to the types. From the perspective of possible world semantics, types provide an "inside-out semantics" where you start from the commonalities (the types) and reason about what objects might be witnesses for the types, rather than starting from a set of possible worlds that have something in common but failing to provide a theory of what the commonality is (apart from saying that a certain sentence is true in all the worlds in the set). Possibly, if you have a theory of situation types, you might try to recover a possible worlds theory by looking at the sets of witnesses of the types. But in order to do this you would need to figure out a way of characterizing situation sufficiently large to count as a world. Normally, in a type theory (or a situation theory, for that matter) there is nothing corresponding to a largest situation. For example, records are finite sets of fields in TTR and can be used to model situations, but for any record it is possible to create a new record by adding an additional field.

### 2.2 Subtyping

Using structured record types allows us to introduce a lattice-like notion of subtyping which is supervenient on this structure. For example, any

situation in which a boy hugs a dog is a situation in which there is a boy. This can be expressed as in (11).

$$(11)\quad \begin{bmatrix} \mathrm{x} & : & \mathit{Ind} \\ \mathrm{c_{boy}} & : & \mathrm{boy(x)} \\ \mathrm{y} & : & \mathit{Ind} \\ \mathrm{c_{dog}} & : & \mathrm{dog(y)} \\ \mathrm{e} & : & \mathrm{hug(x,y)} \end{bmatrix} \sqsubseteq \begin{bmatrix} \mathrm{x} & : & \mathit{Ind} \\ \mathrm{c_{boy}} & : & \mathrm{boy(x)} \end{bmatrix}$$

Here the supertype contains a subset of the fields in the subtype. Exactly which subsets of fields are available as supertypes is regulated by the dependencies among the fields. If we remove a field on which another field depends we have to remove the dependent field as well.

### 2.3 Σ-types

It is often said in type theory that record types are really just a variant notation for Σ-types. Intuitively Σ-types correspond to existential quantification. A Σ-type $(\Sigma x : A)B(\!(x)\!)$ (where we use the notation $B(\!(x)\!)$ to indicate that $B$ depends on $x$) is the type of ordered pairs $\langle a, b\rangle$ where $a : A$ and $b : B(\!(a)\!)$, corresponding to "There is an $A$, $a$, such that $B(\!(a)\!)$". For example, if $A$ is *Dog* and $B$ is *Bark*, the Σ-type $(\Sigma a : \mathit{Dog})\mathit{Bark}(\!(a)\!)$ can be construed as the type of situation in which some dog barks.

We can illustrate the relationship between record types and Σ-types by looking at our running example of a record type corresponding to "some boy hugs some dog" in (12).

$$(12)\quad \begin{bmatrix} \mathrm{x} & : & \mathit{Ind} \\ \mathrm{c_{boy}} & : & \mathrm{boy(x)} \\ \mathrm{y} & : & \mathit{Ind} \\ \mathrm{c_{dog}} & : & \mathrm{dog(y)} \\ \mathrm{e} & : & \mathrm{hug(x,y)} \end{bmatrix} \approx$$

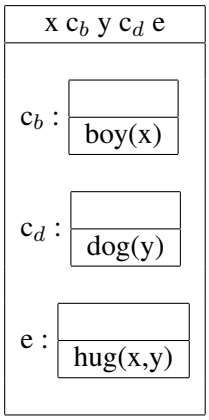
$$\begin{cases} (\Sigma x{:}\mathit{Ind})(\Sigma c_b{:}\mathrm{boy}(x))(\Sigma y{:}\mathit{Ind})(\Sigma c_d{:}\mathrm{dog}(y))\mathrm{hug}(x,y) \\ (\Sigma y{:}\mathit{Ind})(\Sigma c_d{:}\mathrm{dog}(y))(\Sigma x{:}\mathit{Ind})(\Sigma c_b{:}\mathrm{boy}(x))\mathrm{hug}(x,y) \\ \dots \end{cases}$$

Note that as TTR record types are sets of fields there are several Σ-types which intuitively correspond to a single record type. We have represented just two of them in (12). Intuitively, all of these Σ-types are equivalent since they all correspond to "a boy hugs a dog". Note, however, that this equivalence is not directly derivable from the characterization of Σ-types. These Σ-types do not have a witness in common, for example. In TTR we use record types in place of Σ-types.

### 2.4 Discourse representation structures

The labels in record types can play the same role as discourse referents in discourse representation structures (Kamp and Reyle, 1993) in that they can be used to account for anaphora. In particular, the labels associated with situation types seem intuitively related to the discourse referents in segmented discourse representation theory (SDRT, Asher and Lascarides, 2003). Here we will just give an example in (13) of how our running example of a record type can be seen as corresponding to a discourse representation structure.

$$(13)\quad \begin{bmatrix} \mathrm{x} & : & \mathit{Ind} \\ \mathrm{c_{boy}} & : & \mathrm{boy(x)} \\ \mathrm{y} & : & \mathit{Ind} \\ \mathrm{c_{dog}} & : & \mathrm{dog(y)} \\ \mathrm{e} & : & \mathrm{hug(x,y)} \end{bmatrix} \approx$$




### 2.5 Dialogue gameboards/information states

Ginzburg (2012) used TTR record types to represent dialogue gameboards which keep track of the current dialogue state (questions under discussion among other things) according to a dialogue participant. To illustrate this kind of use of record types we present in (14) a preliminary version of the type *InfoState* from Cooper (in prep, Chap. 2) based on the characterization of dialogue gameboards as information states in Larsson (2002).

$$(14)\quad \begin{bmatrix} \text{private:} & \begin{bmatrix} \text{agenda:list}(\mathit{RecType}) \end{bmatrix} \\ \text{shared:} & \begin{bmatrix} \text{latest-utterance:}\mathit{Sign}^* \\ \text{commitments:}\mathit{RecType} \end{bmatrix} \end{bmatrix}$$

Here the information state is divided into two fields: private and shared information, that is private to the particular dialogue participant and shared with the other dialogue participant(s). Among the private information is an agenda, a list of record types, corresponding to types that the dialogue participant plans to realize in upcoming turns in the dialogue.

Among the shared information is the latest utterance, a string of signs in the sense of head driven phrase structure grammar (see 2.6) and commitments, a record type representing what has been agreed on between the dialogue participants for the sake of the dialogue so far.

### 2.6 Feature structures

Record types have also been used in the role of feature structures as used in head-driven phrase structure (HPSG, Sag et al., 2003). To illustrate this, we give, in (15), a version of the basic type *Sign* from Cooper (in prep, Chap. 3).

$$(15)\quad \sigma : \mathit{Sign} \text{ iff } \sigma : \begin{bmatrix} \text{s-event} & : & \mathit{SEvent} \\ \text{syn} & : & \mathit{Syn} \\ \text{cont} & : & \mathit{Cont} \end{bmatrix}$$

where *Syn* is the type in (16).

$$(16)\quad \sigma : \mathit{Sign} \text{ iff } \sigma : \begin{bmatrix} \text{s-event} & : & \mathit{SEvent} \\ \text{syn} & : & \mathit{Syn} \\ \text{cont} & : & \mathit{Cont} \end{bmatrix}$$

Here a sign consists of two fields correponding to the two parts of a Saussurean sign (de Saussure, 1916), speech event and content, and an additional field for syntax, providing a category and a string of signs as daughters. *Sign* is defined as a basic type whose witnesses are witnesses of the record type given rather than directly as that record type. This is because signs can contain other signs (the daughters) and the witnesses of the type *Sign* need to be defined inductively rather than have the type itself be a non-well founded object.

### 2.7 Frames

Record types have also been used to represent frames in the sense of FrameNet (Ruppenhofer et al., 2006) and in the sense of Barsalou (1992)) and the more recent developments in frame theory represented in Löbner et al. (2021). For a discussion of frames in TTR see Cooper (2016) and Cooper (in prep, Chap. 5).

### 2.8 Types as models of concepts, memories and beliefs

We have talked of types as models of propositions. It is natural also to use types to model various kinds of "mental objects". If we think of concepts as types, then we can say that the concept is instantiated if there is a witness for the type. If we think of a memory as a type, then that memory is correct if there is (or was) a witness for the type. It is natural to think of beliefs as propositions and therefore, in our terms, types. A belief is true, then, if there is a witness for the type. Such a view demands that we think of how types could be represented in the brain. Some preliminary suggestions are made by Cooper (2019), arguing that we should consider types as implemented in the brain by patterns of neural activation (rather than as neural architecture). Thinking of mental objects as types, gives the types a dual nature. They can be used to classify the world as in the standard view of types but now in addition they can be used to classify mental states in terms of which types are represented in the mental states.

Let us consider how this idea might look in a little more detail. (For a fuller discussion see Cooper, in prep, Chap. 6.) We treat long term memory (or beliefs) as a type: a type of how the world would be if your memory is correct. An agent's, $a$, long term memory, $T_{\text{ltm}(a)}$, might be characterized as in (17).

$$(17)\quad T_{\text{ltm}(a)} = \begin{bmatrix} \text{id}_1 & : & \begin{bmatrix} \text{x:}\mathit{Ind} \\ \text{e:boy(x)} \end{bmatrix} \\ \text{id}_2 & : & \begin{bmatrix} \text{x:}\mathit{Ind} \\ \text{e:dog(x)} \end{bmatrix} \\ \text{id}_3 & : & \left[\text{e:hug}(\Uparrow\text{id}_1\text{.x}, \Uparrow\text{id}_2\text{.x})\right] \\ \cdots & & \end{bmatrix}$$

Here we have only specified the first three fields of what would probably be a very large type. The '$\Uparrow$'s in the field labelled '$\text{id}_3$' are notational sugar to indicate that the path being referred to is not within the immediate record type in which the path notation occurs but one record type higher up.

Thinking of memory as a record type in this way is similar to thinking of memory as a large mental file (Recanati, 2012) with many subfiles. On this view, the basic intuition about belief, is that $a$ believes $T$ (a type functioning as a proposition), that is, the type 'believe($a$, $T$)' is witnessed ("true"), just in case $T_{\text{ltm}(a)} \sqsubseteq T$. This says that $a$ believes $T$ just in case any way the world could be that is consistent with $a$'s memory would be of type $T$.

However, this basic intuition is not technically quite sufficient to do the job that we need. Suppose that $T$ is our running example of a record type, repeated in (18).

$$(18)\quad \begin{bmatrix} x & : & \textit{Ind} \\ c_{boy} & : & boy(x) \\ y & : & \textit{Ind} \\ c_{dog} & : & dog(y) \\ e & : & hug(x,y) \end{bmatrix}$$

Intuitively, we would want to say that $a$ *does* believe $T$ if $T$ is (18) and $a$'s long term memory is (17). But (17) is *not* a subtype of (18). In fact the two types have no witnesses in common because the labelling required by the two types is different. However, if we *relabel* (18) as indicated in (19) the appropriate subtype relation will hold.

(19) $x \rightsquigarrow id_1.x$, $c_{boy} \rightsquigarrow id_1.e$, $y \rightsquigarrow id_2.x$, $c_{dog} \rightsquigarrow id_2.e$, $e \rightsquigarrow id_3.e$

The relabelling here replaces paths in a type with new paths and thus is able to significantly alter the way in which types are structured, as in this case.

Thus we refine our characterization of what it means for $a$ to believe $T$ by saying that the type 'believe($a$,$T$)' is witnessed just in case there is a relabelling, $\eta$, of $T$ such that $T_{\mathrm{ltm}(a)} \sqsubseteq \eta(T)$. An important consequence of this is that if 'believe($a$,$T$)' is witnessed, so is 'believe($a$,$T'$)' where $T'$ is a relabelling of $T$. That is, labelling is irrelevant for identifying beliefs, even though the labelling is important for other purposes such as identifying anaphora DRT-style.

## 3 Semantic objects *vs* languages

Let us remind ourselves of the central goal of Montague's original semantic project. It was to show that there is in principle no difference between natural languages and formal languages defined by logicians in respect of the fact that it is possible to provide a model theoretic semantics defined on the syntax of natural languages (without first having to translate them into a formal language and characterize a model theoretic semantics for that). This is made explicit in the paper 'English as a Formal Language' (EFL) included in Montague (1974). In 'Universal Grammar' (UG) (also included in Montague, 1974) Montague present a rigorous framework showing how we can use a formal language to represent model theoretic objects and guarantee that translating natural language into this formal language induces a model theoretic semantics defined on the natural language syntax. He does this by composing a homomorphism from the natural language syntactic algebra to the formal language syntactic algebra with a homomorphism from the formal language syntactic algebra to an algebra of semantic objects. This composed homomorphism is from the NL syntactic algebra to the algebra of semantic objects. 'The Proper Treatment of Quanitification in Ordinary English' (PTQ) in (Montague, 1974) is an instance of the framework in UG. Translation into intensional logic is used to induce a model theoretic semantics defined directly on the syntax of English. This makes the presentation much easier to read than the explicit direct interpretation of English syntax into model theory in EFL. The direct interpretation of natural language syntax into the model theory is essential to Montague's original claim that natural languages are formal languages.

TTR introduces structured objects (in the sense that we have discussed) into the realm of semantic objects which play the role of Montague's model theoretic objects and eschews an intermediate language between the natural language syntax and the semantic objects. In this sense TTR adheres to Montague's original project as we have presented it here. There is, however, something puzzling about introducing structured semantic objects in this way: they begin to take on the kind of structure you might expect in syntactic expressions of a formal language. An example, of this is ptypes such as 'hug($b$,$d$)' as discussed earlier. In TTR we also have conjunctive types ($T_1 \wedge T_2$), disjunctive types ($T_1 \vee T_2$) and negative types ($\neg T$). While record types do not have the kind of structure we normally see in a standard logic they do nevertheless have similar structures to those of feature matrices used in syntactic and phonological description.

The construction and inference operations we need to describe and relate structured objects like this seem to take on the syntactic nature of corresponding operations used in proof theory. TTR is not alone in this. It always seems to happen when structured objects are introduced into the semantic domain. Examples from the past are situation semantics (Barwise and Perry, 1983) and logical atomism (Russell, 1918, 1924). Given the normal assumption that model theory models aspects of the world, many find it problematic that the world takes on the structure of a language in this way and for this reason, perhaps, think that a traditional possible worlds semantics is more realistic — despite the intractability of possible worlds and the problems with intensionality.

There are alternatives to introducing structured

objects among the objects in the semantic domain. One of these is to take a radical proof theoretic approach to semantics. According to this we think of semantic theory as providing a mapping from natural language to a proof theoretic language. There may, or may not, be a model theory associated with this language. If there is a model theory it is more concerned with the metalogical study of the proof theoretic language rather than a central component of the semantic theory for natural language. Semantics is seen as primarily involving a correct account of inference rather than associating natural language expressions with the right kind of semantic objects. Perhaps most of the work on a Martin-Löf style type theoretic approach to natural language semantics takes some version of this approach. While inference is undeniably central to semantics and lies at the heart of the motivation for the semantic objects associated with natural language expressions in a model theoretic approach, such a purely proof theoretic approach to inference appears to give up any hope of building a semantic theory which is related to how we perceive and interact with the world.

A second alternative is to introduce an intermediate semantic representation language between natural language syntax and the model theory. An example of this is the use of a discourse representation structure (DRS) language in the early versions of Discourse Representation Theory (Kamp and Reyle, 1993). The use of a Chomskyan logical form together with a formal semantics is another example of this strategy. The point of such theories is that the intermediate representation language introduces structure which is not present in the natural language but which appears to be necessary to facilitate semantic interpretation. Such an intermediate language, therefore, does not follow the discipline set out in Montague's UG and is thus not eliminable in the way that Montague's intensional logic is in PTQ. In effect the argument is that the intermediate language is a necessary part of the theory precisely because it does not meet the conditions inlvolving homomorphisms which is set up in UG. The claim that an intermediate language is necessary is, of course, interesting and non-trivial, but we should be clear that it is abandoning a central tenet of Montague's original project, namely that there is no significant difference between natural languages and formal languages in that they can both have a model theory defined recursively on their syntactic structure. It seems like we cannot give a semantics for natural language after all — we first have to translate it to another language which is suitable for model theoretic interpretation.

Here is a question for both of these alternatives: if we have to translate natural language into a formal language, $L$, in order to give a semantics for the natural language, why is it that we have not evolved to speak $L$ rather than the natural language? Perhaps we can see Montague's insistance on giving a semantics directly on the structure of the natural language as marking him out as an early pioneer of natural logic (van Benthem, 1986, Chapter 6) albeit from a model theoretic rather than a proof theoretic perspective.

Why is it, then, that if we incorporate the kind of structured objects we need into our semantic universe, then these objects appear to take on aspects of structure similar to that of a language? I would like to turn this question around. Perhaps it is not that the objects take on aspects of structure of the language but rather that the language takes on aspects of the structure in terms of which we perceive the world. In TTR we think of the types as providing structured relations between objects in the world, independently of language. This view seems not unrelated to the relational interpretation of quantum theory (Rovelli, 2021) in which the world is structured in terms of observable relations. It seems attractive to say that our languages in certain respects reflect the structure of the world as we perceive it.

This raises the question as to whether the difference between a proof theoretic approach and a model theoretic approach with structured objects is one of philosophical perspective rather than a matter of empirical analysis. One might think that the interesting questions lie not so much in whether you choose a model theoretic or a proof theoretic approach but rather in which kinds of structure you need in order to achieve an adequate account of inference in natural language. This seems to be a reasonable claim.[2] However, there are some reasons which seem to make working with semantic objects preferable to working with expressions in a language.

One reason is the very general one that using semantic objects helps us not lose sight of the fact that the project involves accounting for interaction

[2] And one that I have made a number of times in an ongoing conversation with Ruth Kempson over the past thirty years or so.

with the world, for example, that we need to be talking about the individuation of objects in the real world in addition to making sure that certain expressions stand in appropriate inferential relations. Another reason is that using semantic objects keeps us honest about the exact nature of the structure we are proposing. It can sometimes be easy to write down expressions without being precise about the nature of the structure they encode, for example, whether variables are being used as variables over objects or variables over variables or whether the absence of parentheses is a notational abbreviation or an indication that parentheses are not present in the expression.

A more substantial reason perhaps is that using semantic languages can impose unnecessary or unwanted linguistic structure without us realizing that this is happening. We will take record types as an example of this. Consider a record type as in (20).

(20) $\begin{bmatrix} \ell_1 & : & T_1 \\ \ell_2 & : & T_2 \end{bmatrix}$

In general in the type theory community this record type would be notated as (21).

(21) $\{\ell_1 : T_1; \ell_2 : T_2\}$

As an object it is natural to think of this record type as a *set*, (22), as we have done in this paper.

(22) $\{\langle \ell_1, T\rangle, \langle \ell_2, T_2\rangle\}$

If the record type is a set of fields, then the order of the fields does not matter. On the other hand, if we think of the record type as an expression in a language, then it is natural to think of the fields as ordered. This means that there are two expression record types corresponding the one object record type as in (23).

(23) a. $\{\ell_1 : T_1; \ell_2 : T_2\}$
b. $\{\ell_2 : T_2; \ell_1 : T_1\}$

In general, then, thinking of the record types as expressions leads us into a considerable (and possibly undesirable) multiplication in the number of available record types. An argument for the ordering when thinking of record types as expressions in a language is that if $T_2$ depends on the field $\ell_1 : T_1$ then the $\ell_2$-field must be ordered after the $\ell_1$-field. This is a convention which is standardly followed in proof-theoretically based approaches to type theory. But it is just a convention on the order in which things are written down. Consider the two alternative conventions represented in (24).

(24) a. "Let $n$ be a natural number. Consider $\mathrm{succ}(n)\ldots$"
b. "Consider $\mathrm{succ}(n)$, where $n$ is a natural number…"

When we think of the type as a set it becomes clear that the relevant order involves the order in which the fields are added to the set in constructing it. We can only add a dependent field to a record type which already contains the field on which it depends. This is made clear in the definition of dependent record types given in Cooper (2012, in prep). This does not require us to think of the record type itself as an ordered set.

This might seem like a rather abstract discussion which does not seem to have significant consequences for actual semantic analysis. But note that this discussion points to the difference between record types and $\Sigma$-types discussed in 2.3, where it did have consequences for the inferences we can make.

## 4 Conclusion

In this paper we have discussed what it means to be a structured semantic object and the uses to which structured semantic objects can be put in TTR. In the previous section we discussed the relationship between using structured semantic objects and expressions in a language and suggested that we are presented with a three-way choice in building a semantic theory:

- importing proof theoretic techniques into the model theory
- going entirely proof theoretic
- have an intermediate language between natural language syntax and model theory (and thereby giving up on Montague's project of providing a semantics directly for natural languages)

I have indicated my preference for the first option. Whatever your choice, it does seem that some kind of structured objects or representations are necessary in order to be able to give an adequate semantics for natural languages.

## Acknowledgments

The research reported in this paper was supported by a grant from the Swedish Research Council (VR project 2014-39) for the establishment of the Centre

for Linguistic Theory and Studies in Probability (CLASP) at the University of Gothenburg.

## References


Nicholas Asher and Alex Lascarides. 2003. *Logics of conversation*. Cambridge University Press.

Lawrence W. Barsalou. 1992. Frames, concepts, and conceptual fields. In A. Lehrer and E. F. Kittay, editors, *Frames, fields, and contrasts: New essays in semantic and lexical organization*, pages 21–74. Lawrence Erlbaum Associates, Hillsdale, NJ.

Jon Barwise. 1989. *The Situation in Logic*. CSLI Publications, Stanford.

Jon Barwise and John Perry. 1983. *Situations and Attitudes*. Bradford Books. MIT Press, Cambridge, Mass.

Johan van Benthem. 1986. *Essays in Logical Semantics*. Springer Netherlands.

Robin Cooper. 2012. Type Theory and Semantics in Flux. In Ruth Kempson, Nicholas Asher, and Tim Fernando, editors, *Handbook of the Philosophy of Science*, volume 14: Philosophy of Linguistics, pages 271–323. Elsevier BV. General editors: Dov M. Gabbay, Paul Thagard and John Woods.

Robin Cooper. 2016. Frames as Records. In Annie Foret, Glyn Morrill, Reinhard Muskens, Rainer Osswald, and Sylvain Pogodalla, editors, *Formal Grammar: 20th and 21st International Conferences FG 2015, Barcelona, Spain, August 2015, Revised Selected Papers FG 2016, Bozen, Italy, August 2016, Proceedings*, number 9804 in LNCS, pages 3–18. Springer.

Robin Cooper. 2019. Representing Types as Neural Events. *Journal of Logic, Language and Information*, 28(2):131–155.

Robin Cooper. in prep. From perception to communication: An analysis of meaning and action using a theory of types with records (TTR). Draft available from `https://sites.google.com/site/typetheorywithrecords/drafts`.

Robin Cooper and Jonathan Ginzburg. 2015. Type theory with records for natural language semantics. In Shalom Lappin and Chris Fox, editors, *The Handbook of Contemporary Semantic Theory*, second edition, pages 375–407. Wiley-Blackwell.

Gottlob Frege. 1918/1919. Der Gedanke. Eine logische Untersuchung. *Beiträge zur Philosophie des deutschen Idealismus*, 1:58–77.

Jonathan Ginzburg. 2012. *The Interactive Stance: Meaning for Conversation*. Oxford University Press, Oxford.

Hans Kamp and Uwe Reyle. 1993. *From Discourse to Logic*. Kluwer, Dordrecht.

Staffan Larsson. 2002. *Issue-based Dialogue Management*. Ph.D. thesis, University of Gothenburg.

Sebastian Löbner, Thomas Gamerschlag, Tobias Kalenscher, Markus Schrenk, and Henk Zeevat, editors. 2021. *Concepts, Frames and Cascades in Semantics, Cognition and Ontology*. Springer International Publishing.

Per Martin-Löf. 1984. *Intuitionistic Type Theory*. Bibliopolis, Naples.

Richard Montague. 1974. *Formal Philosophy: Selected Papers of Richard Montague*. Yale University Press, New Haven. Ed. and with an introduction by Richmond H. Thomason.

Bengt Nordström, Kent Petersson, and Jan M. Smith. 1990. *Programming in Martin-Löf's Type Theory*, volume 7 of *International Series of Monographs on Computer Science*. Clarendon Press, Oxford.

François Recanati. 2012. *Mental Files*. Oxford University Press.

Carlo Rovelli. 2021. *Helgoland*. Penguin Books Ltd (UK).

Josef Ruppenhofer, Michael Ellsworth, Miriam R.L. Petruck, Christopher R. Johnson, and Jan Scheffczyk. 2006. FrameNet II: Extended Theory and Practice. Available from the FrameNet website.

Bertrand Russell. 1918. The Philosophy of Logical Atomism. *The Monist*. Reprinted in (Russell, 1956).

Bertrand Russell. 1924. Logical Atomism. In J. H. Muirhead, editor, *Contemporary British Philosophy*. Routledge. Reprinted in (Russell, 1956).

Bertrand Russell. 1956. *Logic and Knowledge: Essays 1901–1950*. George Allen & Unwin. Edited by Robert C. Marsh.

Ivan A. Sag, Thomas Wasow, and Emily M. Bender. 2003. *Syntactic Theory: A Formal Introduction*, 2nd edition. CSLI Publications, Stanford.

Ferdinand de Saussure. 1916. *Cours de linguistique générale*. Payot, Lausanne and Paris. edited by Charles Bally and Albert Séchehaye.

Anil Seth. 2021. *Being You: a New Science of Consciousness*. Faber and Faber.

Philip Wadler. 2015. Propositions as Types. *Communications of the ACM*, 58(12):75–84.