TTR for Natural Language Semantics 2

Robin Cooper, Jonathan Ginzburg

TTR for Natural Language Semantics*

Robin Cooper[1] and Jonathan Ginzburg[2]

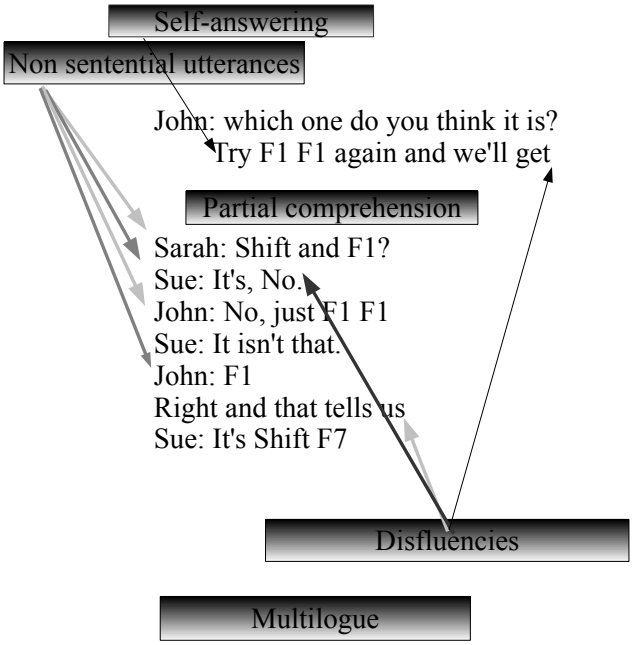
[1] University of Gothenburg cooper@ling.gu.se
[2] Université Paris-Diderot and LabEx-EFL, Sorbonne Paris-Cité
yonatan.ginzburg@univ-paris-diderot.fr

* This work was supported in part by Vetenskapsrådet project 2009-1569, Semantic analysis of interaction and coordination in dialogue (SAICD), by the Lab(oratory of)Ex(cellence)-EFL (ANR/CGI), and by the Disfluency, Exclamations, and Laughter in Dialogue (DUEL) project within the *projets franco-allemand en sciences humaines et sociales* funded by the ANR and the DFG. We are grateful for comments to the participants in three courses we taught in which we presented a version of this material: Type Theory with Records for Natural Language Semantics, NASSLLI, Austin, Texas, 18th – 22nd June, 2012; An introduction to semantics using type theory with records, ESSLLI, Opole, Poland, 13th – 17th Aug, 2012; and Semantics using type theory with records, Gothenburg, 10th – 12th June, 2013. We are grateful to Liz Coppock for comments on an earlier draft of this chapter. Finally, we would like to thank Chris Fox for his very penetrating and careful comments on the first submitted draft.
A draft chapter for the Wiley-Blackwell *Handbook of Contemporary Semantics — second edition*, edited by Shalom Lappin and Chris Fox. This draft formatted on 3rd April 2015.

1 Introduction

Given the state of the art, a simple **actual** conversation such as (1)[2] still constitutes a significant challenge to formal grammar of just about any theoretical flavour.

(1)



As we note in the diagram above, this little dialogue involves a variety of theoretically difficult phenomena: it involves three rather than two participants, is hence a *multi(-party dia)logue*; it features *disfluencies*, a variety of

[2] The conversation occurs in the block G4K of the British National Corpus (BNC). Henceforth, the notation '(BNC,xyz)' refers to the block xyz from the BNC.

types of *non-sentential utterances*, *partial comprehension*, and *self answering*. Making sense of all these phenomena in a systematic way is a challenge undertaken in the TTR–based dialogue framework KoS (Ginzburg, 2012). While we will not have the space to develop a detailed analysis of this example, by the end of the paper we will have sketched a toolbox on the basis of which disfluencies, non-sentential utterances, partial comprehension self, answering, and multilogue can be explicated. A key ingredient to this is a theory of the structure and evolution of *dialogue game-boards*(DGBs), the publicised component of the conversationalists' information states. This, in turn, presupposes both means of developing semantic and grammatical ontologies to explicate notions such as propositions, questions, and utterances.

There are, nonetheless, a number of well established paradigms for doing just that and the obvious question to ask is: why develop a distinct framework? We will illustrate throughout the paper intrinsic problems for frameworks such as possible worlds semantics and typed-feature structure (TFS)–based approaches:

- **Semantic ontology**: Why not a possible worlds–based approach? There are well known problems for this strategy that revolve around its coarseness of grain. These are often ignored (folk assumption: '. . . the attitudes are difficult and primarily a philosophical problem . . . ') Whether or not this is true we point to the problems posed by negation which cannot be brushed off so easily.
- **syntax-semantics interface**: Why is a TFS-based approach to a syntax-semantics interface, as in frameworks such as Head-driven Phrase Structure Grammar (HPSG) (Sag *et al.* (2003)) and in Sign-based Construction Grammar (Michaelis (2009)) insufficient? Here again, there are well known problems (lack of proper binding, functions) and these can be solved in standard λ-calculus based approaches. We will point to issues that are difficult to the latter such as clarification interaction.

Our claim is that TTR enables a uniform theory of grammar, semantics, and interaction that can tackle such problems, while allowing one to maintain past insights (emanating from Montague Semantics and Discourse Representation Theory) and also, we think, future directions (e.g. probabilistic semantics).

This article is structured as follows: the basics of TTR are described in section 2. Subsequently, in sections 3–5 we use this to sketch fundamental notions of grammar, semantic ontology, and dialogical interaction. These are eventually illustrated in more detail in sections 6–8, which deal with meta-communicative interaction, negation, quantification, and, more briefly, non sentential utterances and disfluencies.

2 A theory of types and situations

2.1 Type theory and perception

In classical model theoretic semantics (Montague, 1973, 1974) there is an underlying type theory which presents an ontology of basic classes of objects such as, in Montague's type theory, entities, truth values, possible worlds and total functions between these objects. Here we will make use of a *rich type theory* inspired by the work of Martin-Löf (1984) and much subsequent work on this kind of type theory in computer science. For a recent example relating to natural language see Luo (2011). Ranta (this volume) gives important background on Martin-Löf's type theory.

In a rich type theory of the kind we are considering there are not only types for basic ontological categories but also types corresponding to categories of objects such as *Tree* or types of situations such as *Hugging of a dog by a boy*. A fundamental notion of this kind of type theory is that of a *judgement* that an object (or situation) a is of type T, in symbols, $a : T$. In our view judgements are involved in perception. In perceiving an object we assign it a type. The type corresponds to what Gibson (1986) (and following him in their theory of situation semantics, Barwise & Perry, 1983) would call an *invariance*. In order to perceive objects as being of certain types, agents must be *attuned* to this invariance or type. We take this to mean that the type corresponds to a certain pattern of neural activation in the agent's brain. Thus the types to which a human is attuned may be quite different from those to which an insect is attuned. A bee landing on a tree does not, presumably, perceive the tree in terms of the same type *Tree* that we are attuned to.

2.2 TTR: Type theory with records

The particular type theory we will discuss here is TTR which is particular variant of Type Theory with Records. The most recent published reference which gives details is Cooper (2012). An earlier treatment is given in Cooper (2005b), and Cooper (2005c) discusses its relation to various semantic theories. Here we will give a less detailed formal treatment of the type theory than in the first two of these references. We start by characterizing a *system of basic types* as a pair consisting of a non-empty set of types, **Type**, and a function, A, whose domain is **Type** and which assigns to each type in **Type** a (possibly empty) set which does not overlap with **Type**. We say that *a is of type* T (in **Type**), $a : T$, *according to* $\langle \mathbf{Type}, A\rangle$ just in case $a \in A(T)$. In general we will think of basic types as corresponding to basic ontological categories. The basic type we will use in this section is *Ind* for individuals.

We will use complex types for types of situations, inspired by the notion of situation in Barwise & Perry (1983). The simplest complex type of situation is constructed from a *predicate* together with some appropriate arguments to the predicate. Consider, for example, the type of situation where a boy called

Bill (whom we will represent by b) hugs a dog called Dinah, (represented by d). The type of situation in which Bill hugs Dinah will be constructed from the predicate 'hug' together with the arguments b and d. This type is represented in symbols as hug(b,d). Here we are treating 'hug' as a predicate which has *arity* $\langle Ind, Ind\rangle$, that is, it requires two individuals as arguments. Sometimes we may allow predicates to have more than one arity, that is they may allow different configurations of arguments. In this case we say that the predicate is *polymorphic*.[3] Types like this which are constructed with predicates we will call *ptypes*. A system of types containing ptypes, that is, a *system of complex types*, will be an extension of a system of basic types $\langle \mathbf{BType}, A\rangle$, $\langle \mathbf{Type}, \mathbf{BType}, \mathbf{PType}, \langle A, F\rangle\rangle$ where **PType** is a set of ptypes constructed from a particular set of predicates and arities associated with them by combining them with all possible arguments of appropriate types according to the type system and F is a function whose domain is **PType** which assigns a (possibly empty) set of situations to each ptype. The set **Type** includes both **BType** and **PType**.

This gives us a system of types which will allow us types of situations where particular individuals are related to each other. However, we want to be able to characterize more general types of situation than this, for example, the type of situations where some boy hugs a dog, that is, the type of any "boy hugs dog" situation. There are a number of ways to characterize such more general types in type theory. In TTR we use record types. The type of situation where a boy hugs a dog could be the record type in (2).

(2)
$$\begin{bmatrix} \mathrm{x} & : Ind \\ \mathrm{c_{boy}} & : \mathrm{boy(x)} \\ \mathrm{y} & : Ind \\ \mathrm{c_{dog}} & : \mathrm{dog(y)} \\ \mathrm{e} & : \mathrm{hug(x,y)} \end{bmatrix}$$

This record type consists of five *fields* each of which consists of a *label* (such as 'x' or '$\mathrm{c_{dog}}$') and a type (such as *Ind* or 'dog(y)'). Each field is an ordered pair of a label and a type and a record type is a set of such fields each of which have a distinct label. We use labels like 'x' and 'y' for fields introducing individuals and labels like 'c_{pred}' for fields introducing types which are ptypes with the predicate *pred* representing constraints or conditions (hence 'c') on objects in other fields. We will often use the label 'e' for the type representing the main event, such as hugging.

A record of this type is a set of fields (i.e. order is unimportant) with labels and objects such that no two fields have the same label, there is a field with each of the labels in the record type and the object in the field is of the type in the corresponding field in the record type. Note that there can be more fields in the record with labels not mentioned in the record type. A record of the type in (2), that is, a witness for this type, will be one of the form in (3).

[3] This introduces one kind of polymorphism into the system. We will also introduce some polymorphism in the typing.

(3)
$$\begin{bmatrix} \mathrm{x} & = a \\ \mathrm{c_{boy}} & = s_1 \\ \mathrm{y} & = b \\ \mathrm{c_{dog}} & = s_2 \\ \mathrm{e} & = s_3 \\ \vdots & \end{bmatrix}$$

where:

$a : \mathit{Ind}$
$s_1 : \mathrm{boy}(a)$
$b : \mathit{Ind}$
$s_2 : \mathrm{dog}(b)$
$s_3 : \mathrm{hug}(a, b)$

If the type (2) is non-empty there will be a boy and a dog such that the boy hugs the dog. Thus (2) could be used to represent the content of *a boy hugs a dog*. That is, we use it to play the role of a proposition in other theories. (Later we will introduce a more complex notion of proposition which builds on such types.)

Let r be a record of the form (3). We will refer to the objects in the fields using the notation $r.\ell$ where ℓ is some label in the record. Thus $r.\mathrm{x}$ will be a, $r.\mathrm{c_{boy}}$ will be s_1 and so on. We will allow records to be objects in fields. Thus we can have records within records as in (4).

(4)
$$\begin{bmatrix} f = \begin{bmatrix} \mathit{ff} & = a \\ \mathit{gg} & = b \end{bmatrix} \\ g = c \end{bmatrix} \\ g = \begin{bmatrix} h = \begin{bmatrix} g = a \\ h = d \end{bmatrix} \end{bmatrix}$$

We can extend the dot notation above to refer to *paths* in a record, that is sequences of labels which will lead from the top of a record down a value within the record. Let r be (4). Then we can use paths to denote various parts of the record as in (5).

(5)

a. $r.f = \begin{bmatrix} f = \begin{bmatrix} \mathit{ff} & = a \\ \mathit{gg} & = b \end{bmatrix} \\ g = c \end{bmatrix}$

b. $r.g.h = \begin{bmatrix} g = a \\ h = d \end{bmatrix}$

c. $r.f.f.\mathit{ff} = a$

Technically, we have cheated a little in the presentation of record types. 'boy(x)', 'dog(y)' and 'hug(x,y)' are not technically ptypes since 'x' and 'y' are labels, not individuals as required by the arities of these predicates. What we mean by this notation is the ptype we can construct by substituting whatever individuals occur in the 'x' and 'y' fields of the record we are checking to see

whether it belongs to the type. Thus the ptypes will be different depending on which record you are checking. The official notation for this record type makes this more explicit by introducing functions from individuals to ptypes and pairing them with a list of path names indicating where in the record one should look for the arguments to the functions, as in (6).[4]

(6)
$$\begin{bmatrix} \mathrm{x} & : \mathit{Ind} \\ \mathrm{c_{boy}} & : \langle \lambda v{:}\mathit{Ind} \ . \ \mathrm{boy}(v), \langle \mathrm{x} \rangle \rangle \\ \mathrm{y} & : \mathit{Ind} \\ \mathrm{c_{dog}} & : \langle \lambda v{:}\mathit{Ind} \ . \ \mathrm{dog}(v), \langle \mathrm{y} \rangle \rangle \\ \mathrm{e} & : \langle \lambda v_1{:}\mathit{Ind} \ \lambda v_2{:}\mathit{Ind} \ . \ \mathrm{hug}(v_1, v_2), \\ & \qquad \langle \mathrm{x,y} \rangle \rangle \end{bmatrix}$$

There is good reason to use this more complex notation when we deal with more complex record types which have record types embedded within them. However, for the most part we will use the simpler notation as it is easier to read. Functions from objects to types, *dependent types*, will play an important role in what we have to say below.

In record types we will frequently make use of *manifest fields*[5] A manifest field $\left[\ell{=}a{:}T\right]$ is a convenient notation for $\left[\ell{:}T_a\right]$ where T_a is a singleton type whose only witness is a. Singleton types are introduced by the clauses in (7).

(7)

a. If $a : T$ then T_a is a type.
b. $b : T_a$ iff $b = a$

2.3 Subtyping

The notion of subtype in TTR plays a central inferential role within the system. T_1 is a subtype of T_2 ($T_1 \sqsubseteq T_2$) just in case for all assignments to basic types it is the case that if $a : T_1$ then $a : T_2$. For more discussion of this notion see Cooper (2012).

2.4 Function types

We introduce function types as in (8).

(8)

a. If T_1 and T_2 are types, then so are $(T_1 \to T_2)$ and $(T_1 \to_c T_2)$
b. $f : (T_1 \to T_2)$ iff f is a function with domain $\{a \mid a : T_1\}$ and range included in $\{a \mid a : T_2\}$
c. $f : (T_1 \to_c T_2)$ iff $f : (T_1 \to T_2)$ and there is some $a : T_1$ such that if $b : T_2$ then $f(b) = a$

[4] Here we use the λ-notation for functions which is discussed in Section 2.4.
[5] This notion was introduced in Coquand *et al.* (2004).

This means that f is a total function from objects of type T_1 to objects of type T_2. In (8c) f is required to be a constant function. A function is associated with a graph, that is, a set of ordered pairs, as in the classical set theoretical model of a function. As in set theory we let functions be identified by the graphs, that is, for functions f_1, f_2, if $\mathrm{graph}(f_1) = \mathrm{graph}(f_2)$ then $f_1 = f_2$. We also require that for each graph whose domain (i.e. left projection) is the set of witnesses of a type and whose range (i.e. right projection) is included in the set of witnesses of another type there is a function which has this graph. This makes the existence of a function of type $(T_1 \rightarrow T_2)$ correspond to a universal quantification, "for everything of type T_1 there is something of type T_2". Finally we stipulate that types $(T_1 \rightarrow T_2)$ and T_1 are incompatible. That is, you cannot have something which belongs to a function type and the type which characterizes the domain of the function. As a consequence, functions cannot apply to themselves. This is one way of avoiding paradoxes which can arise when we allow functions to apply to themselves.

We introduce a notation for functions which is borrowed from the λ-calculus as used by Montague (1973). We let functions be identified by sets of ordered pairs as in the classical set theoretic construction of functions. Let $O[v]$ be the notation for some object of our type theory which uses the variable v and let T be a type. Then the function $\lambda v : T \,.\, O[v]$ is to be the function identified by $\{\langle v, O[v]\rangle \mid v : T\}$. For example, the function $\lambda v{:}\mathit{Ind} \,.\, \mathrm{run}(v)$ is identified by the set of ordered pairs $\{\langle v, \mathrm{run}(v)\rangle \mid v : \mathit{Ind}\}$.

Note that if f is the function $\lambda v{:}\mathit{Ind} \,.\, \mathrm{run}(v)$ and $a{:}\mathit{Ind}$ then $f(a)$ (the result of applying f to a) is 'run(a)'. Our definition of function-argument application guarantees what is called β-equivalence in the λ-calculus. We allow both function types and dependent record types and we allow dependent record types to be arguments to functions. We have to be careful when considering what the result of applying a function to a dependent record type should be. Consider the simple example in (9).

(9) $\lambda v_0 : \mathit{RecType}\ (\left[\mathrm{c}_0{:}v_0\right])$

What should be the result of applying this function to the record type in (10)?

(10) $$\begin{bmatrix} \mathrm{x} & : \mathit{Ind} \\ \mathrm{c}_1 & : \langle \lambda v_1 : \mathit{Ind}(\mathrm{dog}(v_1)),\ \langle \mathrm{x} \rangle \rangle \end{bmatrix}$$

Given normal assumptions about function application the result would be (11).

(11) $$\left[\mathrm{c}_0 : \begin{bmatrix} \mathrm{x} & : \mathit{Ind} \\ \mathrm{c}_1 & : \langle \lambda v_1 : \mathit{Ind}\ (\mathrm{dog}(v_1)),\ \langle \mathrm{x} \rangle \rangle \end{bmatrix}\right]$$ (incorrect!)

But this would be incorrect. In fact it is not a well-formed record type since 'x' is not a path in it. Instead the result should be (12).

(12) $$\left[\mathrm{c}_0 : \begin{bmatrix} \mathrm{x} & : \mathit{Ind} \\ \mathrm{c}_1 & : \langle \lambda v_1 : \mathit{Ind}\ (\mathrm{dog}(v_1)),\ \langle \mathrm{c}_0.\mathrm{x} \rangle \rangle \end{bmatrix}\right]$$

Here the path from the top of the record type is specified. However, in the abbreviatory notation we write just 'x' when the label is used as an argument and interpret this as the path from the top of the record type to the field labelled 'x' in the local record type. Thus we will write (13)

(13) $\begin{bmatrix} \mathrm{x} & : \mathit{Ind} \\ \mathrm{c}_1 & : \mathrm{dog(x)} \end{bmatrix}$

(where the 'x' in 'dog(x)' signifies the path consisting of just the single label 'x') and (14)

(14) $\left[\mathrm{c}_0 : \begin{bmatrix} \mathrm{x} & : \mathit{Ind} \\ \mathrm{c}_1 & : \mathrm{dog(x)} \end{bmatrix} \right]$

(where the 'x' in 'dog(x)' signifies the path from the top of the record type down to 'x' in the local record type, that is, 'c_0.x').[6]

Note that this adjustment of paths is only required when a record type is being substituted into a position that lies on a path within a resulting record type. It will not, for example, apply in a case where a record type is to be substituted for an argument to a predicate such as when applying the function (15)

(15) $\lambda v_0 : \mathit{RecType}\ (\left[\mathrm{c}_0 : \mathrm{appear}(v_0)\right])$

to (16)

(16) $\begin{bmatrix} \mathrm{x} & : \mathit{Ind} \\ \mathrm{c}_1 & : \langle \lambda v : \mathit{Ind}\ (\mathrm{dog}(v)),\ \langle \mathrm{x} \rangle \rangle \\ \mathrm{c}_2 & : \langle \lambda v : \mathit{Ind}\ (\mathrm{approach}(v)),\ \langle \mathrm{x} \rangle \rangle \end{bmatrix}$

where the position of v_0 is in an "intensional context", that is, as the argument to a predicate and there is no path to this position in the record type resulting from applying the function. Here the result of the application is (17)

(17) $\left[\mathrm{c}_0 : \mathrm{appear}(\begin{bmatrix} \mathrm{x} & : \mathit{Ind} \\ \mathrm{c}_1 & : \langle \lambda v : \mathit{Ind}\ (\mathrm{dog}(v)),\ \langle \mathrm{x} \rangle \rangle \\ \mathrm{c}_2 & : \langle \lambda v : \mathit{Ind}\ (\mathrm{approach}(v)),\ \langle \mathrm{x} \rangle \rangle \end{bmatrix}) \right]$

with no adjustment necessary to the paths representing the dependencies.[7] (Note that 'c_0.x' is not a path in this record type.)

Suppose that we wish to represent a type which requires that there is some dog such that it appears to be approaching (that is a *de re* reading). In the abbreviatory notation we might be tempted to write (18)

(18) $\begin{bmatrix} \mathrm{x} & : \mathit{Ind} \\ \mathrm{c}_1 & : \mathrm{dog(x)} \\ \mathrm{c}_0 & : \mathrm{appear}(\left[\, \mathrm{c}_2 : \mathrm{approach(x)} \,\right]) \end{bmatrix}$ (incorrect!)

[6] This convention of representing the path from the top of the record type to the "local" field by the final label on the path is new since Cooper (2012).

[7] This record corresponds to the interpretation of *it appears that a dog is approaching.*

corresponding to (19).

(19) $\begin{bmatrix} \mathrm{x} & : & \mathit{Ind} \\ \mathrm{c}_1 & : & \langle \lambda v{:}\mathit{Ind}\ (\mathrm{dog}(v)), \langle \mathrm{x} \rangle \rangle \\ \mathrm{c}_0 & : & \mathrm{appear}(\left[\mathrm{c}_2 : \langle \lambda v{:}\mathit{Ind}\ (\mathrm{approach}(v)), \langle \mathrm{x} \rangle \rangle \right]) \end{bmatrix}$ (incorrect!)

This is, however, incorrect since it refers to a path 'x' in the type which is the argument to 'appear' which does not exist. Instead we need to refer to the path 'x' in the record type containing the field labelled 'c_0' as in (20).

(20) $\begin{bmatrix} \mathrm{x} & : & \mathit{Ind} \\ \mathrm{c}_1 & : & \langle \lambda v{:}\mathit{Ind}\ (\mathrm{dog}(v)), \langle \mathrm{x} \rangle \rangle \\ \mathrm{c}_0 & : & \langle \lambda v{:}\mathit{Ind}\ (\mathrm{appear}(\left[\mathrm{c}_2 : \mathrm{approach}(v) \right])), \langle \mathrm{x} \rangle \rangle \end{bmatrix}$

In the abbreviatory notation we will use '$\Uparrow$' to indicate that the path referred to is in the "next higher" record type[8] as in (21).

(21) $\begin{bmatrix} \mathrm{x} & : & \mathit{Ind} \\ \mathrm{c}_1 & : & \mathrm{dog}(\mathrm{x}) \\ \mathrm{c}_0 & : & \mathrm{appear}(\left[\mathrm{c}_2 : \mathrm{approach}(\Uparrow \mathrm{x}) \right]) \end{bmatrix}$

2.5 Complex types correspondings to propositional connectives

We introduce complex types corresponding to propositional connectives by the clauses in (22).

(22)

a. If T_1 and T_2 are types then so are $(T_1 \wedge T_2)$, $(T_1 \vee T_2)$ and $\neg T$
b. $a : (T_1 \wedge T_2)$ iff $a : T_1$ and $a : T_2$
c. $a : (T_1 \vee T_2)$ iff $a : T_1$ or $a : T_2$
d. $a : \neg T_1$ iff there is some type T_2 which is incompatible with T_1 such that $a : T_2$

T_1 is *incompatible* with T_2 just in case there is no assignment to basic types such that there is some a such that $a : T_1$ and $a : T_2$. That is, it is impossible for any object to belong to both types. This is a non-classical treatment of negation which we will discuss in Section 7.1.

Occasionally we will need types which are possibly infinite joins of types in order to characterize certain function types. We will represent these using a subscripted $\bigvee$. Thus if T_1 and T_2 are types, then (23) is a type.

(23) $$\bigvee_{X \sqsubseteq T_1} (X \to T_2)$$

Witnessing conditions for (23) are defined by (24).

(24) $f : \bigvee_{X \sqsubseteq T_1} (X \to T_2)$ iff $f : (T \to T_2)$ for some type T such that $T \sqsubseteq T_1$.

[8] This notation is new since Cooper (2012).

As we have record types in our system we will be able to form meets, joins and negations of these types just like any other. When we form the meet of two record types, $T_1 \wedge T_2$ there is always a record type T_3 which is equivalent to $T_1 \wedge T_2$ in the sense that no matter what we assign to our basic types anything which is of $T_1 \wedge T_2$ will be of type T_3 and vice versa. T_3 is defined using the merge operator $\wedge\!\!\!\cdot$. Thus, $T_1 \wedge\!\!\!\cdot\, T_2$ is the *merge* of the two types T_1,T_2. If at least one of the two types is not a record type it is identical with the meet $T_1 \wedge T_2$. The basic idea of *merge* for record types is illustrated by the examples in (25).

(25)

a. $\left[\text{f}{:}T_1\right] \wedge\!\!\!\cdot \left[\text{g}{:}T_2\right] = \begin{bmatrix}\text{f}{:}T_1\\ \text{g}{:}T_2\end{bmatrix}$

b. $\left[\text{f}{:}T_1\right] \wedge\!\!\!\cdot \left[\text{f}{:}T_2\right] = \left[\text{f}{:}T_1 \wedge\!\!\!\cdot\, T_2\right]$

(For a full definition which makes clear what the result is of merging any two arbitrary types, see Cooper, 2012.) Merge corresponds to unification in feature based systems such as HPSG.

In addition to merge we also introduce *asymmetric merge*, $T_1 \boxed{\wedge\!\!\!\cdot}\, T_2$. This is defined like ordinary merge except that in the case where one of the types is not a record type $T_1 \boxed{\wedge\!\!\!\cdot}\, T_2 = T_2$. This notion (which is discussed in detail in Cooper, in prep) is related to that of priority unification (Shieber, 1986).

2.6 Set and list types

We introduce set and list types as in (26).

(26)

a. If T is a type then $\{T\}$ and $[T]$ are types

b. $A : \{T\}$ just in case A is a set and for any $a \in A$, $a : T$

c. $L : [T]$ just in case L is a list and any member, a, of L is such that $a : T$

We will also introduce a type $Poset(T)$ which can be regarded as (27)

(27) $$\begin{bmatrix}\text{set} & : \{T\}\\ \text{rel} & : \{\begin{bmatrix}\text{left} & : T\\ \text{right} & : T\end{bmatrix}\}\\ \text{c}_{po} & : \text{po(rel,set)}\end{bmatrix}$$

where $a : \text{po}(R, S)$ iff $a = \langle R, S\rangle$ and R is a partial order on S, that is, R is a set of pairs of members of S (coded as records with 'left' and 'right' fields as above) and R is reflexive or irreflexive, antisymmetric and transitive.

If $a : T$, $P : Poset(T)$ and $a \notin P.\text{set}$, then $a \oplus P : Poset(T)$ where $a \oplus P$ is the record $r{:}Poset(T)$ such that the clauses in (28) hold.

(28)

a. $r.\text{set}{=}P.\text{set} \cup \{a\}$

b. $r.\text{rel} = P.\text{rel} \cup \{ \begin{bmatrix} \text{left} & = \text{a} \\ \text{right} & = \text{x} \end{bmatrix} \mid x \in P.\text{set} \}$

c. $r.\text{c}_{po} = \langle r.\text{rel}, r.\text{set} \rangle$

2.7 The string theory of events

So far we have talked about situations or events in terms of ptypes or record types which have ptypes in some of their fields. This gives us a rather static view of events and does not give an analysis of the changes that take place as an event unfolds. A single type is rather like a snapshot of an event at one point in its development. In an important series of papers including Fernando (2004, 2006, 2008, 2009); **?**), Tim Fernando has proposed that events should be analyzed in terms of strings of snapshots or observations. In TTR we adapt these ideas by introducing *regular types*: types of *strings* of objects corresponding to the kinds of strings you find in regular languages in formal language theory (Hopcroft & Ullman, 1979; Partee *et al.*, 1990). (29) is an account of the two main kinds of regular types that we will use here where $a^\frown b$ represents the *concatenation* of two objects a and b.

(29)

a. if $T_1, T_2 \in \mathbf{Type}$, then $T_1{}^\frown T_2 \in \mathbf{Type}$
   $a : T_1{}^\frown T_2$ iff $a = x^\frown y$, $x : T_1$ and $y : T_2$

b. if $T \in \mathbf{Type}$ then $T^+ \in \mathbf{Type}$.
   $a : T^+$ iff $a = x_1^\frown \ldots {}^\frown x_n$, $n > 0$ and for i, $1 \leq i \leq n$, $x_i : T$

$T_1{}^\frown T_2$ is the type of strings where something of type T_1 is concatenated with something of type T_2. T^+ is the type of non-empty strings of objects of type T. Suppose for example that we want to represent the type a game of fetch as a game played between a human, a, and a dog, b, involving a stick, c, in which the human picks up the stick, attracts the attention of the dog, and throws the stick, whereupon the dog runs after the stick and picks it up, returning it to the human, after which the cycle can start from the beginning. The type of this event would be (30).

(30) $(\text{pick\_up}(a,c)^\frown \text{attract\_attention}(a,b)^\frown \text{throw}(a,c)^\frown \text{run\_after}(b,c)^\frown \text{pick\_up}(b,c)^\frown \text{return}(b,c,a))^+$

2.8 Inference from partial observation of events

An important fact about our perception of events is that we can predict the type of the whole event when we have only perceived part of the event. Thus if we see a human and a dog playing with a stick and we see the human pick up the stick and attract the dog's attention we might well predict that the type of the whole event is one of playing fetch. We can represent this prediction by a function, as in (31).

(31) $\lambda r{:}\begin{bmatrix} \mathrm{x} & : \mathit{Ind} \\ \mathrm{c}_{\mathrm{human}} & : \mathrm{human(x)} \\ \mathrm{y} & : \mathit{Ind} \\ \mathrm{c}_{\mathrm{dog}} & : \mathrm{dog(y)} \\ \mathrm{z} & : \mathit{Ind} \\ \mathrm{c}_{\mathrm{stick}} & : \mathrm{stick(z)} \\ \mathrm{e} & : \mathrm{pick\_up(x,z)}^\frown\mathrm{attract\_attention(x,y)} \end{bmatrix}$
$(\left[\mathrm{e} : \mathrm{play\_fetch}(r.\mathrm{x},r.\mathrm{y})\right])$

Notice that this function is what we have called a dependent type, that is, it takes an object (in this case the observed situation) and returns a type (in this case the type of the predicted situation). Notice that this ability to predict types of situations on the basis of partial observations is not particular to humans. The dog realizes what is going on and probably starts to run before the human has actually thrown the stick. However, in the Section 3 we will suggest that humans build on this ability in their perception and analysis of speech events.

3 Grammar in TTR

In Section 2 we suggested that an important capability that agents have is the prediction of the type of a complete event on the basis of a partial observation of an event. We suggested that functions from observed situations to predicted situation type (a kind of dependent type) can be used in modelling this, taking the example of the game of fetch. Very similar inferences are involved in the perception of linguistic events, though there are also some important differences. In the case of the game of fetch the predicted type is a type of situation which you could in principle perceive completely. In the example we gave you are inferring the nature of the event as it will develop later in time. The case of linguistic perception is rather more abstract. We are inferring types which may hold simultaneously with what we have observed and the predicted event types may be of events that are not directly perceivable. Thus we are able to perceive events belonging to phonological or phonetic types but from these we infer types relating to syntactic and semantic structure whose instances are not directly perceivable. It is this kind of reasoning about abstract objects which seems so important to human linguistic ability. Nevertheless the fundamental mechanism is the same: we are mapping from an observation to a type of something unobserved.

Grammar rules involve a prediction on the basis of a string of linguistic events. Thus they are functions of the form (32).

(32) $\lambda s : T_1^\frown \ldots ^\frown T_n(T)$

where the T_i and T are *sign types*, which, as we will see below, are types which have both a directly perceivable and a non-directly perceivable component. Thus grammar rules are functions from strings of linguistic events to a type of a single linguistic event. An example would be the observation of a string consisting of a noun-phrase event followed by a verb-phrase event and predicting that there is a sentence event, that is, what is normally written in linguistic formalisms as the phrase-structure rule S $\rightarrow$ NP VP.

Sign types correspond to the notion of sign in HPSG (Sag *et al.*, 2003). The type *Sign* could be thought of as (33).[9]

(33)
$$\begin{bmatrix} \text{s-event} : \mathit{SEvent} \\ \text{synsem} : \begin{bmatrix} \text{cat} & : \mathit{Cat} \\ \text{constits} & : \{\mathit{Sign}\} \\ \text{cont} & : \mathit{Cont} \end{bmatrix} \end{bmatrix}$$

Here we use 'synsem' ("syntax and semantics") as a field corresponding to both syntactic and semantic information, although this, and also what follows below, could be adjusted to fit more closely with other versions of HPSG. However, for technical reasons having to do with recursion (ultimately signs

[9] For more detailed discussion of the grammar discussed here and below see Cooper (2012).

may be contained within signs), we have to define *Sign* as a basic type which meets the condition (34).

(34) r:*Sign* iff r: $\begin{bmatrix} \text{s-event} & : \textit{SEvent} \\ \text{synsem} & : \begin{bmatrix} \text{cat} & : \textit{Cat} \\ \text{constits} & : \{\textit{Sign}\} \\ \text{cont} & : \textit{Cont} \end{bmatrix} \end{bmatrix}$

We have introduced three new types here: *SEvent*, the type of speech events; *Cat*, the type of categories and *Cont*, the type of semantic contents. We will take each of these in turn and return to the 'constits'-field (for "constituents") in synsem later.

A minimal solution for the type *SEvent* is (35).

(35) $\begin{bmatrix} \text{phon} & : \textit{Phon} \\ \text{s-time} & : \textit{TimeInt} \\ \text{utt}_{\text{at}} & : \text{uttered\_at(phon, s-time)} \end{bmatrix}$

Here we have introduced the types *Phon*, phonology, and *TimeInt*, time interval, which we will further specify below. A more detailed type for *SEvent* might be (36).

(36) $\begin{bmatrix} \text{e-time} & : \textit{TimeInt} \\ \text{e-loc} & : \textit{Loc} \\ \text{sp} & : \textit{Ind} \\ \text{au} & : \textit{Ind} \\ \text{phon} & : \textit{Phon} \\ \text{e} & : \text{utter(sp,phon,au,e-time,e-loc)} \end{bmatrix}$

where we have in addition fields for event location, speaker and audience. This corresponds more closely to the kind of information we normally associate with speech act theory (Searle, 1969). However, this type may be too restrictive: more than one person may be in the audience; more than one speaker may collaborate on a single speech event, as is shown by work on split utterances (Purver *et al.*, 2010). For present purposes it will be sufficient to use the simpler type (35) for speech events.

We will take the type *Phon* to be the type of a non-empty string of phoneme utterances, that is $\textit{Phoneme}^{+}$. We could use phonetic symbols to represent types of individual phoneme utterances. For example u : `h` would mean that u is an utterance of the phoneme `h` (the phoneme being modelled as a TTR type). u : `h‿æy` would mean that u is an utterance of the phoneme string which we denote in orthography by 'hi'. It is not our intention to give a detailed account of phonology here and we will represent this string type using the orthography as `hi`. Note that `hi` is a subtype of *Phon*.

We define the type *TimeInt*, for time interval, to be (37).

(37) $\begin{bmatrix} \text{start} & : \textit{Time} \\ \text{end} & : \textit{Time} \\ \text{c}_{<} & : \text{start<end} \end{bmatrix}$

where *Time* is a basic type whose witnesses are time points and $<$ is a predicate (here used in infix notation) which requires that its first argument is ordered before its second argument.

The 'constits'-field in synsem if for the set of constituents (including all constituents, not just daughters (immediate constituents)).

In Section 5 we will extend the definition of *Sign* to include a field for a dialogue game board.

4 A theory of abstract entities

An ontology including abstract entities—including entities such as propositions, questions, and outcomes is a necessary ingredient for accounts of illocutionary acts such as assertion, querying, and commanding, as well as of attitude reports. Building on a conception articulated 30 years earlier by Austin (1961), Barwise & Etchemendy (1987) developed a theory of propositions in which a proposition is a structured object $prop(s, \sigma)$, individuated in terms of a situation s and a situation type σ. Given the ':' relation between situations and their types there is a a straightforward notion of truth and falsity:

(38)

a. $\text{prop}(s, \sigma)$ is true iff $s : \sigma$ (s *is of type* σ).
b. $\text{prop}(s, \sigma)$ is false iff $s \nmid \sigma$ (s *is not of type* σ).

A detailed such ontology extending the original situation semantics ontology was developed in Ginzburg & Sag (2000). This approach has subsequently been developed in TTR in works such as Ginzburg (2011, 2012). We start by discussing how to add propositions into TTR.

For many purposes the type theory already developed has entities that could be identified with Austinian propositions, an identification frequently assumed in past work in type theory via the slogan *propositions as types*.

Cooper (2005b) develops the former in which a proposition p is taken to be a record type. A witness for this type is a situation. On this strategy, a witness is not directly included in the semantic representation. Indeed, record types *are* competitive in such a role: they are sufficiently fine-grained to distinguish identity statements that involve distinct constituents. (39a) would correspond to the record type in (39c), whereas (39b) to the record type in (39d)). Moreover, in this set up substitutivity of co-referentials (39e) and cross-linguistic equivalents ((39e), the Hebrew equivalent of (39a)) can be enforced:

(39)

a. Enescu is identical with himself.
b. Poulenc is identical with himself.
c. $\left[\text{c} : \text{Identical(enesco, enesco)}\right]$
d. $\left[\text{c} : \text{Identical(poulenc, poulenc)}\right]$
e. He is identical with himself.
f. Enesku zehe leacmo.

A situational witness for the record type could also be deduced to explicate cases of event anaphora, as in (40); indeed, a similar strategy is invoked when in an analysis of nominal anaphora in Ginzburg (2012):

(40)

a. A: Jo and Mo got married yesterday. It was a wonderful occasion.

b. A: Jo's arriving next week. B: No, that's happening in about a month.

Nonetheless, here we develop an explicitly Austinian approach, where the situational witness is directly included in the semantic representation. The original Austinian conception was that s is a situation deictically indicated by a speaker making an assertion[10]—teasing out the semantic difference between implicit and explicit witnesses is a difficult semantic task. The Austinian approach is important for negation (see section 7.1). Explicitly Austinian propositions can also play a role in characterizing the communicative process: in section 6 we will show that *locutionary propositions* individuated in terms of an utterance event u_0 as well as to its grammatical type T_{u_0} allows one to simultaneously define update and clarification potential for utterances. In this case, there are potentially many instances of distinct locutionary propositions, which need to be differentiated on the basis of the utterance token—minimally any two utterances classified as being of the same type by the grammar.

Assuming we adopt an explicitly Austinian approach, then on the current account the type of propositions is the record type (41a). The correspondence with the situation semantics conception is quite direct. We can define truth conditions as in (41b).

(41)

a. $Prop =_{def} \begin{bmatrix} \text{sit} & : Rec \\ \text{sit-type} & : RecType^\dagger \end{bmatrix}$

b. A proposition $p = \begin{bmatrix} \text{sit} & = s_0 \\ \text{sit-type} & = ST_0 \end{bmatrix}$ is true iff $s_0 : ST_0$

Here the type $RecType^\dagger$ is a basic type which denotes the type of records types closed under meet, join and negation. That is, we require:

(1) if T:*RecType*, then T:$RecType^\dagger$
(2) if T_1, T_2:$RecType^\dagger$, then $T_1 \wedge T_2$, $T_1 \vee T_2$, $\neg T_1$:$RecType^\dagger$
(3) Nothing is of type $RecType^\dagger$ except as required above.

If p:*Prop* and p.sit-type is $T_1 \wedge T_2$ ($T_1 \vee T_2$, $\neg T$) we say that p is the conjunction (disjunction) of $\begin{bmatrix} \text{sit} & = p.\text{sit} \\ \text{sit-type} & = T_1 \end{bmatrix}$ and $\begin{bmatrix} \text{sit} & = p.\text{sit} \\ \text{sit-type} & = T_2 \end{bmatrix}$ (the negation of $\begin{bmatrix} \text{sit} & = p.\text{sit} \\ \text{sit-type} & = T \end{bmatrix}$). This means that Austinian propositions are not closed under conjunction and disjunction. You can only form the conjunction and disjunction of Austinian propositions which have the same situation. If p_1 and p_2 are Austinian propositions such that $p_1.\text{sit} = p_2.\text{sit}$, we say that p_1 *entails* p_2 just in case $p_1.\text{sit-type} \sqsubseteq p_2.\text{sit-type}$.

A subtype of *Prop* that will be important below is the type of locutionary propositions *LocProp*. Locutionary propositions are Austinian propositions about utterances. *LocProp* is defined as follows:

[10] One could also construe s as *evidence* (a body of knowledge, a database) which provides support (or otherwise) for the type σ.

$$LocProp =_{def} \begin{bmatrix} \text{sit} & : Sign \\ \text{sit-type} & : RecType^{\dagger} \end{bmatrix}$$

4.1 Questions

Given the existence of Austinian-like propositions and a theory of λ-abstraction given to us by existence of functional types, it is relatively straightforward to develop a theory of questions as propositional abstracts in TTR. Extensive motivation for the view of questions as propositional abstracts is provided in Ginzburg (1995); Ginzburg & Sag (2000)—TTR contributes to this by providing an improved notion of simultaneous, restricted abstraction, as we will see shortly.

A (basic, non-compound) question will be a function from records into propositions. As such, questions are automatically part of the type theoretic ontology. Let us start by considering some very simple examples of interrogatives and their TTR representations. (42) exemplifies the denotations (contents) we can assign to a unary and a binary *wh*-interrogative. We use r_{ds} here to represent the record that models the described situation in the context. The meaning of the interrogative would be a function defined on contexts which provide the described situation and which return as contents the functions given in (42). The unary question ranges over instantiations by persons of the proposition "x runs in situation r_{ds}". The binary question ranges over pairs of persons x and things y that instantiate the proposition "x touches y in situation r_{ds}":

(42)

a. who ran $\mapsto$

$$\lambda r{:}\begin{bmatrix} \text{x:Ind} \\ \text{rest:person(x)} \end{bmatrix} \left(\begin{bmatrix} \text{sit} & = r_{ds} \\ \text{sit-type} & = [\text{c:run}(r.\text{x})] \end{bmatrix}\right)$$

b. who touched what $\mapsto$

$$\lambda r{:}\begin{bmatrix} \text{x:Ind} \\ \text{rest1:person(x)} \\ \text{y:Ind} \\ \text{rest2:thing(y)} \end{bmatrix} \left(\begin{bmatrix} \text{sit} & = r_{ds} \\ \text{sit-type} & = [\text{c:touch}(r.\text{x},r.\text{y})] \end{bmatrix}\right)$$

What of polar questions? Ginzburg & Sag (2000) proposed that these are 0-ary abstracts, though the technical apparatus involved in explicating this notion in their framework based on non-well-founded set theory was quite complex. TTR, however, offers a simple way to explicate 0-ary abstraction. If we think of a unary abstract as involving a domain type with one field for an individual and a binary abstract as one whose domain type contains two such fields etc, then by analogy the domain type of a 0-ary type would simply be the empty record type [] (that is, the type *Rec* of records).[11] This makes a 0-ary abstract a constant function from the universe of all records . (43) exemplifies this:

[11] This is the type all records satisfy, since it places no contraints on them.

(43)

Did Bo run $\mapsto$

$\lambda r{:}Rec(\begin{bmatrix} \text{sit} & = r_{ds} \\ \text{sit-type} & = \begin{bmatrix} \text{c} : \text{run(bo)} \end{bmatrix} \end{bmatrix})$

The fact that questions individually are part of the type theoretic world is not the end of the story. For various linguistic tasks (e.g. specifying the selectional requirements of verbs like 'ask', 'wonder', and 'investigate'), and for various dialogical tasks (e.g. the formulation of various conversational rules) one needs to appeal to a type *Question* (see the chapter on questions, Wiśniewski (this volume).). This means that we need to have a characterization of this type within TTR. One such characterization is given in Ginzburg (2012); a more recent and, arguably, more constructive proposal can be found in Ginzburg *et al.* (2014a). Here we offer a somewhat simpler characterization.The domain of a question (polar or *wh*) is always characterized by a subtype of *RecType*. Thus we define the type *Question* by (44).

(44) $\quad Question =_{def} \bigvee_{X \sqsubseteq RecType} (X \rightarrow Prop)$

The type of polar questions, *PolQuestion*, is given in (45).

(45) $\quad PolQuestion =_{def} (Rec \rightarrow_c Prop)$

That is, polar questions are constant functions from situations (records) to propositions as discussed in Ginzburg (2012).

Answerhood is one of the essential testing grounds for a theory of questions. Abstracts can be used to *underspecify* answerhood. This is important given that NL requires a variety of answerhood notions, not merely exhaustive answerhood or notions straightforwardly definable from it. Moreover, as with questions, answerhood needs to be explicable within type theory. This is because answerhood figures as a constituent relation of the lexical entries of resolutive verbs[12] and in rules regulating felicitous responses in dialogue management (see section 5.). For current purposes this means that we need to be able to define notions of answerhood as types.

There are a number of notions of answerhood that are of importance to dialogue. One relates to coherence: any speaker of a given language can recognize, independently of domain knowledge and of the goals underlying an interaction, that certain propositions are *about* or *directly concern* a given question. We will call this *Aboutness*. The simplest notion of answerhood we can define on the basis of an abstract is one we will call, following Ginzburg & Sag (2000), *simple answerhood*. In order to this we will use the following notion:

> A proposition p is an *instantiation* of a question q just in case there is some r in the domain of q such that $q(r) = p$

[12] For more detailed discussion see Ginzburg & Sag (2000, Chapter 3, section 3.2; Chapter 8, section 8.3.).

(46) α is a *simple answer* to q iff α is an instantiation of q or the negation of an instantiation of q.

Given these definitions it is straightforward to show:

(47)

a. If q is an n-ary question of type $(T \rightarrow Prop)$ and α is a simple answer to q then there is some $r : T$ such that α is $q(r)$ or $\neg q(r)$.

b. In particular, if q is the polar question $\lambda r{:}[](p)$ and α is a simple answer to q then α is either p or $\neg p$.

Simple answerhood covers a fair amount of ground. But it clearly underdetermines aboutness. On the polar front, it leaves out the whole gamut of answers to polar questions that are weaker than p or $\neg p$ such as conditional answers 'If r, then p' (e.g. 48a) or weakly modalized answers 'probably/possibly/maybe/possibly not p' (e.g. (48b)). As far as wh-questions go, it leaves out quantificational answers (48c–g), as well as disjunctive answers. These missing class of propositions, are pervasive in actual linguistic use:

(48)

a. Christopher: Can I have some ice-cream then?
Dorothy: you can do if there is any. (BNC, KBW)

b. Anon: Are you voting for Tory?
Denise: I might. (BNC, KBU, slightly modified)

c. Dorothy: What did grandma have to catch?
Christopher: A bus. (BNC, KBW, slightly modified)

d. Rhiannon: How much tape have you used up?
Chris: About half of one side. (BNC, KBM)

e. Dorothy: What do you want on this?
Andrew: I would like some yogurt please. (BNC, KBW, slightly modified)

f. Elinor: Where are you going to hide it?
Tim: Somewhere you can't have it.(BNC, KBW)

g. Christopher: Where is the box?
Dorothy: Near the window. (BNC, KBW)

One straightforward way to enrich simple answerhood is to consider the relation that emerges by closing simple answerhood under disjunction. Ginzburg (1995); Ginzburg & Sag (2000) show that aboutness as defined in (49) seems to encompass the various classes of propositions exemplified in (48).

(49) p is About q iff p entails a disjunction of simple answers to q.

Answerhood in the 'aboutness' sense is clearly distinct from a highly restricted notion of answerhood, that of being a proposition that *resolves* or constitutes *exhaustive information* about a question. This latter sense of answerhood, which is restricted to true propositions, has been explored in great

detail in the formal semantics literature, since it is a key ingredient in explicating the behaviour of interrogatives embedded by resolutive predicates such as 'know', 'tell' and 'discover'. We will not discuss this here but refer the reader to Ginzburg (2012).

Many queries are responded to with *a query*. A large proportion of these are *clarification requests*, to be discussed in section 6. But in addition to these, there are query responses whose content directly addresses the question posed, as exemplified in (50):

(50)

a. A: Who murdered Smith? B: Who was in town?
b. A: Who is going to win the race? B: Who is going to participate?
c. Carol: Right, what do you want for your dinner?
   Chris: What do you (pause) suggest? (BNC, KBJ)
d. Chris: Where's mummy?
   Emma: Mm?
   Chris: Mummy?
   Emma: What do you want her for? (BNC, KBJ)

There has been much work on relations among questions within the framework of *Inferential Erotetic Logic* (IEL) (see e.g. Wiśniewski (2001, 2003) and Wiśniewski (this volume)), yielding notions of q(uestion)–*implication*. From this a natural hypothesis can be made about such query responses, as in (51a). A related proposal, first articulated by Carlson (1983), is that they are constrained by the semantic relations of *dependence*, or its converse *influence*.

(51) a. q_2 can be used to respond to q_1 if q_2 influences q_1.
b. q_2 Influences q_1 iff any proposition p such that p Resolves q_2, also satisfies p entails r such that r is About q_1.

Its intuitive rationale is this: discussion of q_2 will necessarily bring about the provision of information about q_1. The actual characterization of query responses is complex, both empirically and theoretically. For a detailed account using TTR see Łupkowski & Ginzburg (2014).

5 Interaction on dialogue gameboards

On the approach developed in KoS the analysis of dialogue is formulated at a level of information states, one per conversational participant. Each information state consists of two 'parts', a private part and the dialogue gameboard that represents information that arises from publicized interactions. For recent psycholinguistic evidence supporting this partition see Brown-Schmidt *et al.* (2008).

Information states are records of the type given in (52a). For now we focus on the dialogue gameboard, various aspects of which are exploited in the toolbox used to account for the phenomena exemplified in our initial example from the BNC. The type of dialogue gameboards is given in (52b). The *spkr,addr* fields allow one to track turn ownership. *Facts* represents conversationally shared assumptions. *Moves* and *Pending* represent, respectively, lists of moves that have been grounded or are as yet ungrounded. *QUD* tracks the questions currently under discussion.

(52)

a. $\mathit{TotalInformationState}\ (\mathit{TIS}) =_{def} \begin{bmatrix} \text{dialoguegameboard} & : \mathit{DGBType} \\ \text{private} & : \mathit{Private} \end{bmatrix}$

b. $\mathit{DGBType} =_{def}$

$$\begin{bmatrix} \text{spkr} & : \mathit{Ind} \\ \text{addr} & : \mathit{Ind} \\ \text{utt-time} & : \mathit{TimeInt} \\ \text{c-utt} & : \text{addressing(spkr,addr,utt-time)} \\ \text{Facts} & : \{\mathit{Prop}\} \\ \text{Pending} & : [\mathit{LocProp}] \\ \text{Moves} & : [\mathit{LocProp}] \\ \text{QUD} & : \text{poset}(\mathit{Question}) \end{bmatrix}$$

Our job as dialogue analysts is to construct a theory that will explain how conversational interactions lead to observed conversational states of type *DGBType*. Let us consider how an initial conversational state looks, that is the state as the first utterance of the dialogue is made. Initially no moves have been made and no issues introduced, so a dialogue gameboard will be of the type in (53):

(53)

$$\begin{bmatrix} \text{spkr} & : \mathit{Ind} \\ \text{addr} & : \mathit{Ind} \\ \text{utt-time} & : \mathit{TimeInt} \\ \text{c-utt} & : \text{addressing(spkr,addr,utt-time)} \\ \text{Facts=\{\}} & : \{\mathit{Prop}\} \\ \text{Pending=[]} & : [\mathit{LocProp}] \\ \text{Moves=[]} & : [\mathit{LocProp}] \\ \text{QUD=\{\}} & : \text{poset}(\mathit{Question}) \end{bmatrix}$$

692 This allows us to construct a type corresponding to a lexical entry for a
693 greeting word such as 'hi', as in (54). Here we assume that the definition of
694 the type *Sign* in Section 3 has been modified to include a field for a dialogue
695 game board:

$$Sign =_{def} \begin{bmatrix} \text{s-event} & : & SEvent \\ \text{synsem} & : & \begin{bmatrix} \text{cat} & : & Cat \\ \text{cont} & : & cont \end{bmatrix} \\ \text{dgb} & : & DGBType \end{bmatrix}$$

697 This represents an extension of the Saussurean notion of sign where we not
698 only take account of the signifier ('s-event') and the signified ('synsem') but
699 also the context in which the signification takes place (here represented by
700 'dgb').

701 (54)
$$Sign \wedge \begin{bmatrix} \text{s-event:} \begin{bmatrix} \text{phon:}\texttt{hi} \\ \text{s-time:}\, TimeInt \end{bmatrix} \\ \text{synsem:} \begin{bmatrix} \text{cat=interj:}\, Cat \\ \text{cont=} \begin{bmatrix} \text{sit} & =r_{ds} \\ \text{sit-type=} & \begin{bmatrix} \text{e:greet(}\Uparrow\text{dgb.spkr,} \\ \Uparrow\text{dgb.addr,}\Uparrow\text{dgb.utt-time)} \end{bmatrix} \end{bmatrix} : Prop \end{bmatrix} \\ \text{dgb:} \begin{bmatrix} \text{spkr:}\, Ind \\ \text{addr:}\, Ind \\ \text{utt-time=s-event.s-time:}\, TimeInt \\ \text{moves=[]:[Prop]} \\ \text{qud=\{\}:poset(}\, Question) \end{bmatrix} \end{bmatrix}$$

703 Here, as before in our discussion of questions, r_{ds} is the described situation as
704 determined by the context. The use of '$\Uparrow$' in the 'sit-type'-field is a convenient
705 informal notation for paths occurring in a record type embedded within a
706 larger record type but not lying on a path in that record type. It indicates
707 that the path is to be found in the next higher record type. It clears up an
708 ambiguity that arises because we are using the notation that does not make
709 explicit the dependent types that are being used as discussed on p. 6.

710 How do we specify the effect of a conversational move? The basic units
711 of change are mappings between dialogue gameboards that specify how one
712 gameboard configuration can be modified into another on the basis of dialogue
713 moves. We call a mapping between DGB types a *conversational rule*. The types
714 specifying its domain and its range we dub, respectively, the *pre(conditions)*
715 and the *effects*, both of which are supertypes of the type *DGBType*. A con-
716 versational rule that enables us to explain the effect a greeting, the initial
717 conversational move, has on the DGB is given in (55). It is a record type
718 which contains two fields. The 'pre(condition)'-field is for a dialogue game-
719 board of a certain type and the 'effects'-field provides a type for the updated
720 gameboard. The precondition in this example requires that both Moves and

QUD are empty; the sole effect is to push the proposition associated with *hi* onto the list in the 'moves'-field.

(55)

$$\begin{bmatrix} \text{pre}{:}DGBType\!\!\wedge\!\!\!\cdot \begin{bmatrix} \text{spkr} & : Ind \\ \text{addr} & : Ind \\ \text{utt-time} & : TimeInt \\ \text{moves}{=}[] & : [Prop] \\ \text{qud}{=}\{\} & : \text{poset}(Question) \end{bmatrix} \\ \text{effects}{=} \begin{bmatrix} \text{moves}{=}[\begin{bmatrix} \text{sit} & {=}r_{ds} \\ \text{sit-type}{=} \begin{bmatrix} \text{e:greet(pre.spkr,} \\ \text{pre.addr,pre.utt-time)} \end{bmatrix} \end{bmatrix} \\ \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{|pre.moves]}{:}[Prop] \end{bmatrix} {:}RecType \end{bmatrix}$$

The form for update rules proposed here is thus

(56)
$$\begin{bmatrix} \text{pre} & : T_1 \\ \text{effects}{=}T_2 & : RecType \end{bmatrix}$$

An agent who believes that they have a current state s of type T_1, that is, whose hypothesis about the current state is that it belongs to type T such that $T \sqsubseteq T_1$, can use s to anchor T_2 to obtain $T_2[s]$ and then use asymmetric merge to obtain a type for the new state: $T \boxed{\wedge\!\!\!\cdot} T_2[s]$.

The rule (57) says that given a question q and ASK(A,B,q) being the LatestMove, one can update QUD with q as QUD–maximal.

(57) Ask QUD–incrementation

$$\begin{bmatrix} \text{ques}{:}Question \\ \text{moves-tail}{:}[Prop] \\ \text{pre}{:}DGBType\!\!\wedge\!\!\!\cdot \begin{bmatrix} \text{spkr}{:}Ind \\ \text{addr}{:}Ind \\ \text{moves}{=}[\begin{bmatrix} \text{sit} & = r_{ds} \\ \text{sit-type} & = \begin{bmatrix} \text{e:ask(pre.spkr,} \\ \text{pre.addr,ques)} \end{bmatrix} \end{bmatrix} \\ \qquad\qquad\qquad\qquad\qquad\qquad \text{|moves-tail]} : [Prop] \\ \text{qud:poset}(Question) \end{bmatrix} \\ \text{effects}{=}\big[\, \text{q}{=}\text{ques}{\oplus}\text{pre.qud} : \text{poset}(Question) \,\big]{:}RecType \end{bmatrix}$$

Next we introduce the rule QSPEC. QSPEC can be thought of as a 'relevance maxim': it characterizes the contextual background of reactive queries and assertions. (58) says that if q is QUD–maximal, then subsequent to this the next move is constrained to be *q–specific* (Ginzburg, 2012), that is, either about q (a partial answer) or a question on which q depends. Moreover, this move can be held by either of the speech event participants. The constraint in (58) involves merging a constraint concerning the information about QUD and Moves with a constraint entitled TurnUnderSpec, which merely specifies

that the speaker and addressee of the effects are distinct and drawn from the set consisting of the initial speaker and addressee:

(58) a. QSPEC

$$\begin{bmatrix} \text{pre} : \left[\text{qud.} = \left\langle \text{q, Q} \right\rangle : \text{poset(Question)}\right] \\ \text{effects} : \text{TurnUnderspec} \mathbin{\dot{\wedge}} \\ \left[\begin{array}{l} \text{r} : \text{Prop} \vee \text{Question} \\ \text{R: IllocRel} \\ \text{LatestMove} = \text{R(spkr,addr,r)} : \text{IllocProp} \\ \text{c1} : \text{About(r,q)} \vee \text{Depend(q,r)} \end{array}\right] \end{bmatrix}$$

b. $\text{TurnUnderspec} = \begin{bmatrix} \text{PrevAud} = \left\{\text{pre.spkr,pre.addr}\right\} : \left\{\text{Ind}\right\} \\ \text{spkr} : \quad \text{Ind} \\ \text{c1} : \quad \text{member(spkr, PrevAud)} \\ \text{addr} : \quad \text{Ind} \\ \text{c2:} \quad \text{member(addr, PrevAud)} \\ \wedge\ \text{addr} \neq \text{spkr} \end{bmatrix}$

QSPEC involves factoring out turn taking from the assumption that A asking q involves B answering it. In other words, the fact that A has asked q leaves underspecified who is to address q (first or at all). This is justified by self-answering data such as the initial example we considered in the introduction (1), as well as (59a,b), where the querier can or indeed needs to keep the turn, as well as multi-party cases such as (59c) where the turn is multiply distributed:

(59)

a. Vicki: When is, when is Easter? March, April? (BNC, KC2)
b. Brian: you could encourage, what's his name? Neil. (BNC, KSR)
c. A: Who should we invite? B: Perhaps Noam. C: Martinu. D: Bedrich.

Explicating the possibility of self-answering is one of the requirements for dealing with our initial example (1).

6 Unifying metacommunicative and illocutionary interaction

Establishing that the most recent move has been understood to the satisfaction of the conversationalists, has come to be known as *grounding*, following extensive empirical work by Herb Clark and his collaborators (Clark & Schaefer (1989); Clark & Wilkes-Gibbs (1986); Clark (1996)). One concrete task for a theory of dialogue is to account for the potential for and meaning of acknowledgement phrases, as in (60), either once the the utterance is completed, as in (60a), or concurrently with the utterance as in (60b):

(60)

a. Tommy: So Dalmally I should safely say was my first schooling. Even though I was about eight and a half. Anon 1: Mm. Now your father was the the stocker at Tormore is that right ? (British National Corpus (BNC, K7D)
b. A: Move the train ...
   B: Aha
   A:... from Avon ...
   B: Right
   A:... to Danville. (Adapted from the Trains corpus, Allen *et al.* (1995))

An additional task is to characterize the range of (potential) presuppositions emerging in the aftermath of an utterance, whose subject matter concerns both content and form. This is exemplified in the constructed examples in (61):

(61)

a. A: Did Mark send you a love letter?
b. B: No, though it's interesting **that you refer to Mark/my brother/our friend**
c. B: No, though it's interesting **that you mention sending**
d. B: No, though it's interesting **that you ask a question containing seven words.**
e. B: No, though it's interesting **that the final two words you just uttered start with 'l'**

Developing a semantic theory that can fully accommodate the challenges of grounding is far from straightforward. A more radical challenge, nonetheless, is to explicate what goes on when an addressee cannot ground her interlocutor's utterance. We suggest that this is more radical because it ultimately leads to seemingly radical conclusions of an intrinsic *semantic indeterminacy*: in such a situation the public context is no longer identical for the interlocutors—the original speaker can carry on, blissfully unaware that a problem exists, utilizing a 'grounded context', whereas if the original addressee takes over the context is shifted to one which underwrites a *clarification request*. This

potential context–splitting is illustrated in (62), originally discussed in (Ginzburg (1997)). The data in (62) illustrates that the contextual possibilities for resolving the fragment 'Bo?' are distinct for the original speaker A and the original addressee B. Whereas there is one common possibility, the short answer reading, only B has the two clarification request readings, whereas only A has a self-correction reading, albeit one that probably requires an further elaboration:

(62)

a. A: Who does Bo admire? B: Bo?
   Reading 1 (**short answer**): Does Bo admire Bo?
   Reading 2 (**clausal confirmation**): Are you asking who BO (of all people) admires?;
   Reading 2 (**intended content**): Who do you mean 'Bo'?)
b. A: Who does Bo admire? Bo?
   Reading 1 (**short answer**): Does Bo admire Bo?
   Reading 2 (**self correction**): Did I say 'Bo'?

Clarification requests can take many forms, as illustrated in (63):

(63)

a. A: Did Bo leave?
b. **Wot**: B: Eh? / What? / Pardon?
c. **Explicit (exp)** : B: What did you say? / Did you say 'Bo' /What do you mean 'leave'?
d. **Literal reprise (lit)**: B: Did BO leave? / Did Bo LEAVE?
e. **Wh-substituted Reprise (sub)**: B: Did WHO leave? / Did Bo WHAT?
f. **Reprise sluice (slu)**: B: Who? / What? / Where?
g. **Reprise Fragments (RF)**: B: Bo? / Leave?
h. **Gap**: B: Did Bo . . . ?
i. **Filler (fil)**: A: Did Bo . . . B: Win? (Table I from Purver (2006))

Now, as (64a) indicates, *a priori* ANY sub-utterance is clarifiable, including function words like 'the', as in (64c). While the potential for repetition-oriented clarification interaction clearly applies to all utterances and their parts, it is an open question whether this is true for semantically/pragmatically oriented CRification. For empirical studies on this see Healey *et al.* (2003); Purver *et al.* (2003, 2006).

(64)

a. Who rearranged the plug behind the table?
b. Who? / rearranged?/ the plug? / behind? / the table?
c. A: Is that the shark? B: The? B: Well OK, *a.* (based on an example in the film *Jaws.*)

Integrating metacommunicative interaction into the DGB involves two additions to the picture we have had so far, one minor and one major. The minor addition, drawing on an early insight of Conversation Analysis (see the notion of *side sequence*, Schegloff (2007)), is that repair can involve 'putting aside' an utterance for a while, a while during which the utterance is repaired. The 'pending'-field in the dialogue gameboard is used for this. Note that this field contains a list of locutionary propositions. Most work on (dialogue) context to date involves reasoning and representation solely on a semantic/logical level. But if we wish to explicate metacommunicative interaction, then we cannot limit ourselves in this way.

If p:*LocProp*, the relationship between p.sit and p.sit-type can be utilized in providing an analysis of grounding/CRification conditions:

(65)

a. Grounding: p is true: the utterance type fully classifies the utterance token.
b. CRification: p is false, either because p.sit-type is weak (e.g. incomplete word recognition) or because u is incompletely specified (e.g. incomplete contextual resolution—problems with reference resolution or sense disambiguation).

In principle one could have a theory of CRification based on generating all available CRs an utterance could give rise to. But in practice, as the data in (64) showed us, there are simply too many to be associated in a 'precompiled' form with a given utterance type.

Instead, repetition and meaning–oriented CRs can be specified by means of a uniform class of conversational rules, dubbed *Clarification Context Update Rules (CCURs)* in Ginzburg (2012). Each CCUR specifies an accommodated MaxQUD built up from a sub-utterance u1 of the target utterance, the maximal element of Pending (*MaxPending*). Common to all CCURs is a license to follow up *MaxPending* with an utterance which is *co-propositional* with MaxQUD. (66) is a simplified formulation of one CCUR, Parameter identification, which allows B to raise the issue about A's sub-utterance u: *what did A mean by u?*:

(66) Parameter identification:

$$\begin{bmatrix} \text{max\_pending} : \mathit{LocProp} \\ \text{rst\_pending} : [\mathit{LocProp}] \\ \text{u} : \mathit{Sign} \\ \text{c}_{\text{u}} : \text{member(u,max\_pending.sit.synsem.constits)} \\ \text{latest\_move} : \mathit{LocProp} \\ \text{rst\_moves} : [\mathit{LocProp}] \\ \text{pre} : \mathit{DGBType} \wedge\!\!\!\cdot \left[\text{ spkr } \right] \mathit{Ind} \\ \text{pending} : [\text{max\_pending}|\text{rst\_pending}] : [\mathit{LocProp}] \\ \text{moves} : [\text{latest\_move}|\text{rst\_moves}] : [\mathit{LocProp}] \\ \text{qud} : [\mathit{Question}] \\ \text{effects} : \left[\text{ qud}=[q \mid \text{pre.qud}] : [\mathit{Question}] \right] \end{bmatrix}$$

where q is $\lambda r{:}\left[\text{cont}{:}\mathit{Cont}\right]$ $\left(\left[\text{ e} : \text{mean}(\Uparrow\text{pre.spkr},\Uparrow\text{pre.u},r.\text{cont}) \right]\right)$

Parameter Identification (66) underpins CRs such as (67b–67c) as follow-ups to (67a). We can also deal with corrections, as in (67d), since they address the issue of what A meant by u.

(67) a. A: Is Bo here?
b. B: Who do you mean 'Bo'?
c. B: Bo? (= Who is 'Bo'?)
d. B: You mean Jo.

We have now explicated the basis for partial comprehension in dialogue, relating to the requirements from the initial example (1).

7 Traditional semantic concerns in a dialogue perspective

In this section we will discuss two traditional concerns in semantics, negation and quantification, and show that we get a rather different view of them when we consider dialogue phenomena relating to them.

7.1 Negation

The classical view of negation is that it is a truth functional connective that maps true to false and false to true. In intuitionistic approaches as standardly used in type theory, negative propositions, $\neg p$, are regarded as the type of refutations of p. This leads intuitionistic logic to abandon the principle of bivalence, that propositions are either true or false. On the intuitionistic view it is possible that a proposition p neither has a proof nor a refutation. Thus such a proposition is neither true nor false.

In this section, which contains revised material from Cooper & Ginzburg (2011a,b), we will suggest an alternative view: that negation is used to pick out a negative situation type. It is crucial for this proposal to work that we are able to distinguish between positive and negative types in a way that is not possible on the standard approaches to "truth-value flipping" negation.

Consider the uses of *no* in the (made-up) dialogue in (68) and the glosses given after them in square brackets.

(68)

Child approaches socket with nail
Parent: No. ["Don't put the nail in the socket."]
Do(#n't) you want to be electrocuted?
Child: No. ["I don't want to be electrocuted."]
Parent: No. ["You don't want to be electrocuted."]

The first use of *no* does not relate back to any previous linguistic utterance but rather to an event which is in progress. The parent has observed the first part of the event and predicted a likely conclusion (as in the example of the game of fetch discussed in Section 2). The parent wishes to prevent the completion of the event, that is, make sure that the predicted complete event type is not realized. We claim that the central part of the meaning of negation has to do with the non-realization of some positive situation type (represented by a negative situation type), rather than a switching of truth values as on the classical logical view of negation. We see this again in the second use of *no* in response to the parent's query whether the type *child-wants-to-be-electrocuted* is realized. The child's negative response asserts that the type is not realized. The third utterance of *no* agrees with the previous assertion, namely this asserts agreement that the type is (or should be) empty. A naive application of the classical view of negation as a flipping of truth values might say that *no* always changes the truth-value of the previous assertion. This would make the

wrong prediction here, making the parent disagree with the child. Our view that negation has to do with a negative situation type means that it will be used to disagree with a positive assertion and agree with a negative assertion, which seems to be how negation works in most, if not all, natural languages.

Another important fact about this dialogue is the choice of the parent's question. The positive question is appropriate whereas the negative question would be very strange, suggesting that the child should want to be electrocuted. The classical view of negation as truth value flip has led to a view that positive and negative questions are equivalent (Hamblin, 1973; Groenendijk & Stokhof, 1997). This derives from a view of the contents of questions as the sets of propositions corresponding to their answers. While positive and negative questions do seem to have the same possible answers it appears that the content of the question should involve something more than the set of answers. The distinction between positive and negative questions was noted for embedded questions by Hoepelmann (1983) who noted the examples in (69).

(69)

a. The child wonders whether 2 is even.

b. The child wonders whether 2 isn't even. (There is evidence that 2 is even)

Hoepelmann's observation is that the same kind of inference as we noticed with the negative version of the parent's question about electrocution. That is, there is a suggestion that there is reason to believe the positive, that the type is realized. This kind of inference is not limited to negative questions but seems to be associated with negation in general. Fillmore (1985) notes the examples in (70).

(70)

a. Her father doesn't have any teeth
b. #Her husband doesn't have any walnut shells
c. Your drawing of the teacher has no nose/#noses
d. The statue's left foot has no #toe/toes

The examples marked with # sound strange because they are contrary to our expectations. We in general expect that people have teeth but not walnut shells, a nose but not several noses and several toes but not just a single toe. Fillmore discusses this in terms of frames. We would discuss this in terms of resources we have available. We can, however, create the expectations by raising issues for discussion within the dialogue thus creating the necessary resources locally as in (71).

(71) A: My husband keeps walnut shells in the bedroom.
B: Millie's lucky in that respect. Her husband doesn't have any walnut shells.

This discussion points to a need to distinguish between positive and negative propositions based on positive and negative situation types. We have given the two reasons in (72) for this:

(72)

a. The content of *no* is different depending on whether it is used in a response to a negative or positive proposition
b. The raising of a contrary expectation occurs only with negative assertions or questions

A third reason which has been discussed in the literature (recently by Farkas & Roelofsen ms) is that some languages have different words for *yes* depending on positive and negative propositions. This is illustrated in (73).

(73)

a. A: Marie est une bonne étudiante
   Marie is a good student
   B: Oui / #Si.
   Yes / Yes (she is)
b. A: Marie n'est pas une bonne étudiante
   Marie isn't a good student
   B: #Oui / Si.
   Yes / Yes (she is)

In French the word *oui* is used to agree with a positive proposition and the word *si* is used to disagree with a negative proposition. Similar words exists in other languages such as German (*ja/doch*) and Swedish (*ja/jo*).

How do we know that the distinction between positive and negative propositions is a semantic distinction rather than a syntactic distinction depending on how the propositions are introduced? There are lots of ways of making a negative sentence, by using various negative words such as *not*, *no*, *none*, *nothing*. In French you have there are discontinuous constructions *ne... pas/point/rien* corresponding to "not/not at all/nothing". However, in these constructions the *ne* can be omitted. Thus both of the following are possible: *je n'en sais rien/ j'en sais rien* ("I know nothing about it"). In Swedish there are two words for *not* which are stylistic variants: *inte, ej* The generalization that allows us to recognize all these morphemes or constructions as "negations" is the semantic property they share: namely that they introduce negative propositions.

On the traditional truth-value flipping view of negation it is hard to make this semantic distinction. For example, in a possible worlds semantics a proposition is regarded as a set of possible worlds – the set of worlds in which the proposition is true. On this view the negation of a proposition is the complement of that set of worlds belonging to the proposition. There is no way of distinguishing between "positive" and "negative" sets of possible worlds. However, on a type theoretic approach the distinction can be made in a straightforward manner.

The account of negation we give here is slightly different to that given in Cooper & Ginzburg (2011a,b) and as a consequence the definitions are slightly more elegant and intuitive. We introduce negative types by the clause (74).

(74) If T is a type then $\neg T$ is a type

Because types are intensional we can say that $\neg T$ is distinct not only from T but also from any other type, without worrying that there might be an equivalent type that has the same witnesses. Thus simply by introducing a negative operation on types (represented by $\neg$) we distinguish negative types from positive ones. We can also introduce types of negative types. For example, we can introduce a type $\mathit{RecType}^{\neg}$ such that $T{:}\mathit{RecType}^{\neg}$ iff $T = \neg T'$ and $T'{:}\mathit{RecType}$. We can then define a type $\mathit{RecType}^{\neg\dagger}$ whose witnesses are the closure of the set of negated record types under negation (in a similar manner to our definition of $\mathit{RecordType}^{\dagger}$ on p. 18).

We can characterize witnesses for negative types by: $a : \neg T$ iff there is some T' such that $a : T'$ and T' precludes T. We say that T' *precludes* T iff either (75a) or (75b) holds.

(75)

a. $T = \neg T'$
b. or, T, T' are non-negative and there is no a such that $a : T$ and $a : T'$ for any models assigning witnesses to basic types and ptypes

It follows from these two definitions that (1) $a : \neg\neg T$ iff $a : T$ and that (2) $a : T \vee \neg T$ is *not* necessary (a may not be of type T and there may not be any type which precludes T either). Thus this negation is a hybrid of classical and intuitionistic negation in that (1) normally holds for classical negation but not intuitionistic whereas (2), that is failure of the law of the excluded middle, normally holds for intuitionistic negation but not classical negation.

Nothing in these definitions accounts for the fact that $a : \neg T$ seems to require an expectation that $a : T$. One way to do this is to refine the clause that defines witnesses for negative types:$a : \neg T$ iff there is some T' such that $a : T'$ and T' precludes T *and there is some expectation that* $a : T$. There is some question in our minds of whether this addition should be included here or in some theory of when agents are likely to make judgements. What does it mean for there to "be some expectation"? We would like to relate this to the kind of functions we used to predict completions of events and which we also used for grammar rules, that is to dependent types. Breitholtz (2010); Breitholtz & Cooper (2011) use dependent types to implement Aristotelian enthymemes that is defeasible inference patterns. Such enthymemes could be either general or local context-specific resources that we have available to create expectations.

Finally, let us see how the techniques we have developed here could be combined with Austinian propositions.

The type of negative Austinian propositions can be defined as (76).

(76) $\begin{bmatrix} \text{sit} & : Rec \\ \text{sit-type} & : RecType^{\neg\dagger} \end{bmatrix}$

The type of positive Austinian propositions can be defined as (77).

(77) $\begin{bmatrix} \text{sit} & : Rec \\ \text{sit-type} & : RecType \end{bmatrix}$

Thus we have a clear way of distinguishing negative and positive propositions.

7.2 Generalized quantifiers

Purver & Ginzburg (2004); Ginzburg & Purver (2012); Ginzburg (2012) introduce the Reprise Content Hypothesis (RCH) given in (78).

(78)

a. **RCH (weak)** A fragment reprise question queries a part of the standard semantic content of the fragment being reprised.
b. **RCH (strong)** A fragment reprise question queries exactly the standard semantic content of the fragment being reprised.

They use this to motivate a particular view of the semantics of quantified noun-phrases which is based on witness sets rather than families of sets as in the classical treatment. Cooper (2010, 2013) argues for combining a more classical treatment with their approach. We summarize the argument here.

In terms of TTR, a type corresponding to a "quantified proposition" can be regarded as (79).

(79) $\begin{bmatrix} \text{restr} & : Ppty \\ \text{scope} & : Ppty \\ \text{c}_q & : q(\text{restr,scope}) \end{bmatrix}$

The third field represents a quantificational ptype of the form q(restriction,scope) an example of which would be (80).

(80) $\text{every}(\lambda r{:}[\text{x}{:}Ind]([\text{c}{:}\text{dog}(r.\text{x})]),\ \lambda r{:}[\text{x}{:}Ind]([\text{c}{:}\text{run}(r.\text{x})]))$

That is, 'every' is a predicate which holds between two properties, the property of being a dog and the property of running. As an example, suppose we want to represent the record type which is the content of an utterance of *A thief broke in here last night*. For convenience we represent the property of being a thief as *thief* and the property corresponding to *broke in here last night* as *bihln*. Then the content of the sentence can be (81).

(81) $\begin{bmatrix} \text{restr}{=}\text{'}thief\text{'} & : Ppty \\ \text{scope}{=}\text{'}bihln\text{'} & : Ppty \\ \text{c}_\exists & : \exists(\text{restr,scope}) \end{bmatrix}$

We can relate this proposal back to classical generalized quantifier theory, as represented in Barwise & Cooper (1981). Let the *extension* of a type T, $[\check{}T]$, be the set $\{a \mid a : T\}$, the set of witnesses for the type. Let the *P-extension* of a property P, $[\downarrow P]$, be the set in (82).

(82) $\{a \mid \exists r[r : \left[\text{x:}Ind\right] \wedge r.\text{x} = a \wedge [\check{}P(r)] \neq \emptyset]\}$

That is, the set of objects that have the property. We say that there is a constraint on models such that the type $q(P_1, P_2)$ is non-empty iff the relation q^* holds between $[\downarrow P_1]$ and $[\downarrow P_2]$, where q^* is the relation between sets corresponding to the quantifier in classical generalized quantifier theory. Examples are given in (83).

(83)

a. some(P_1,P_2) is non-empty (that is, "true") just in case $[\downarrow P_1] \cap [\downarrow P_2] \neq \emptyset$
b. every(P_1,P_2) is non-empty just in case $[\downarrow P_1] \subseteq [\downarrow P_2]$.
c. many(P_1,P_2) is non-empty just in case $\mid [\downarrow P_1] \cap [\downarrow P_2] \mid > n$, where n counts as many.

The alternative analysis of generalized quantifiers that Purver & Ginzburg (2004); Ginzburg & Purver (2012); Ginzburg (2012) propose is based on the notion of witness set from Barwise & Cooper (1981). Here we will relate this notion to the notion of a witness for a type, that is something which is of that type. We have not yet said exactly what it is that is of a quantifier ptype $q(P_1, P_2)$. One solution to this is to say that it is a witness set for the quantifier, that is (84).[13]

(84) $a : q(P_1, P_2)$ iff q^* holds between $[\downarrow P_1]$ and $[\downarrow P_2]$ and $a = [\downarrow P_1] \cap [\downarrow P_2]$

This definition relies on the fact that all natural language quantifier relations are conservative (Peters & Westerståhl, 2006), a notion which we can define as in (85).

(85) a quantifier q is *conservative* means q^* holds between $[\downarrow P_1]$ and $[\downarrow P_2]$ just in case q^* holds between $[\downarrow P_1]$ and $[\downarrow P_1] \cap [\downarrow P_2]$ (*every person runs* iff *every person is a person who runs*)

Armed with this we can define the type of *(potential) witness sets for a quantifier relation q and a property P*, $q^\dagger(P)$, that is, witness sets in the sense of Barwise and Cooper as in (86).

(86) $a : q^\dagger(P)$ iff $a \subseteq [\downarrow P]$ and there is some set X such that q^* holds between $[\downarrow P]$ and X

Using these tools we present a modified version of Ginzburg and Purver's proposed analysis of *most students left* in (87), where the 'q-params'-field specifies quantifier parameters and the 'cont'-field specifies the content of the utterance.

(87)
$$\begin{bmatrix} \text{q-params} : & \left[\text{w:most}^\dagger(\text{student})\right] \\ \text{cont} & : \left[\text{c}_q\text{=q-params.w:most(student,left)}\right] \end{bmatrix}$$

[13] This appears to go against the intuition that we have introduced before that ptypes are types of situations. Ultimately we might wish to say that a witness for a quantifier type is a situation containing such a witness set, but we will not pursue this here.

In Cooper (2010) we presented the two analyses as in competition with each other, but we now think that there is advantage to be gained by putting the two together. Our way of combining the two analyses predicts two readings for the noun-phrase *most students*, a referential reading which makes the witness set be a q-parameter in Purver and Ginzburg's analysis and a non-referential reading in which the witness set is incorporated in the content of the NP. These are given in (88).

(88)

a. **referential**

$$\left[\begin{array}{l}\text{q-params}:\left[\begin{array}{ll}\text{restr}_i=\text{student}:Ppty \\ \text{w}_i & :\text{most}^{\dagger}(\text{q-params.restr}_i)\end{array}\right] \\ \text{cont}= \\ \quad \lambda P{:}Ppty \\ \qquad (\left[\begin{array}{ll}\text{scope}=P & :Ppty \\ \text{c}_{\text{most}}=\Uparrow\text{q-params.w}_i & :\text{most}(\Uparrow\text{q-params.restr}_i, \\ & \qquad \text{scope})\end{array}\right]):Quant\end{array}\right]$$

b. **non-referential**

$$\left[\begin{array}{l}\text{q-params}:Rec \\ \text{cont}= \\ \quad \lambda P{:}Ppty \\ \qquad (\left[\begin{array}{l}\text{restr}_i=\text{student}:Ppty \\ \text{w}_i:\text{most}^{\dagger}(\text{restr}_i) \\ \text{scope}=P:Ppty \\ \text{c}_{\text{most}}=\text{w}_i:\text{most}(\text{restr}_i,\text{scope})\end{array}\right]):Quant\end{array}\right]$$

Given these types, what can a clarification address? Our claim is that the clarification must address something for which there is a path in the record type. In addition there appears to be a syntactic constraint that clarifications tend to be a "major constituent", that is a noun-phrase or a sentence, rather than a determiner or a noun. In a referential reading there are three paths available: 'q-params.restr$_i$', 'q-params.w$_i$' and 'cont'. The first of these, the restriction, is dispreferred for syntactic reasons since it is normally expressed by a noun. This leaves the witness and the whole NP content as possible clarifications. However, from the data it appears that the whole content can be expressed focussing either on the restriction or the quantifier relation. For non-referential readings only the whole content path is available.

In (89) we give one example of each kind of clarification from the data that Purver and Ginzburg adduce.

(89)

a. *Witness clarification*

Unknown: And er they X-rayed me, and took a urine sample, took a blood sample. Er, the doctor
Unknown: Chorlton?
Unknown: **Chorlton**, mhm, he examined me, erm, he, he said now they were on about a slide ⟨unclear⟩ on my heart. Mhm, he couldn't find it.

BNC file KPY, sentences 1005–1008

b. *Content clarification with restriction focus*
Terry: Richard hit the ball on the car.
Nick: What car?
Terry: **The car that was going past**.

BNC file KR2, sentences 862–864

c. *Content clarification with quantifier relation focus*
Anon 2: Was it nice there?
Anon 1: Oh yes, lovely.
Anon 2: Mm.
Anon 1: It had twenty rooms in it.
Anon 2: **Twenty rooms?**
Anon 1: **Yes**.
Anon 2: How many people worked there?

BNC file K6U, sentences 1493–1499

Our conclusion is that a combination of the classical approach to generalized quantifiers combined with a modification of the approach suggested by Purver and Ginzburg, adding a field for the witness, provides correct predictions about clarifications. This means that the strong version of the reprise clarification hypothesis is consistent with our analysis, allbeit now with a more complex interpretation of the clarification request than Purver and Ginzburg proposed. The interpretation proposed here involves a combination of the classical approach to generalized quantifiers and the witness approach suggested by Purver and Ginzburg. The clarification itself, however, can address different parts of the content of the clarification request.

8 Grammar in dialogue

8.1 Non Sentential Utterances

The basic strategy adopted in KoS to analyze non sentential utterances (NSUs) is to specify construction types where the combinatorial operations integrate the (surface) denotata of the fragments with elements of the DGB. We have provided one example of this earlier in our lexical entry for 'hi', (54). Another simple example is given in (90), a lexical entry for the word 'yes'.

(90) $$Sign\wedge \begin{bmatrix} \text{s-event} : \left[\text{phon} : \texttt{yes}\right] \\ q_{max} : PolQuestion \\ \text{synsem} : \begin{bmatrix} \text{cat=adv\_ic} & : Cat \\ \text{cont=}q_{max}(r_{ds}) & : Prop \end{bmatrix} \end{bmatrix}$$

Here q_{max} is a maximal element of dgb.qud which is of the type PolQuestion, exemplified in (43). Since q_{max} is of the type PolQuestion, it is a constant function whose domain is the class of all records and its range is a proposition p. Hence the content of this function applied to any record is p. Thus, 'yes' gets as its content the proposition p, intuitively affirming the issue 'whether p' currently under discussion. See Fernández (2006); Ginzburg (2012) for a detailed account of this and a wide range of other more complex NSU types.

8.2 Disfluencies

Disfluencies are ubiquitous and observable in all but the briefest conversational interaction. Disfluencies have been studied by researchers in Conversational Analysis (e.g., Schegloff *et al.* (1977)), in great detail by psycholinguists (e.g., Levelt (1983); Brennan & Schober (2001); Clark & Tree (2002)), and by computational linguists working on speech applications (e.g., Shriberg (1994)). To date, they have mostly been excluded from semantic analysis, primarily because they have been assumed to constitute low level 'noise', without semantic import. In fact, disfluencies participate in semantic and pragmatic processes such as anaphora, conversational implicature, and discourse particles, as illustrated in (91).

(91)

a. Peter was + { well } he was] fired. (Example from Heeman & Allen (1999))
b. A: Because I, [[[any, + anyone,] + any friend,] + anyone] I give my number to is welcome to call me (Example from the Switchboard corpus, Godfrey *et al.* (1992)) (implicature: 'It's not just her friends that are welcome to call her when A gives them her number')
c. From yellow down to brown–NO–that's red. (Example from Levelt (1983))

In all three cases, the semantic process is dependent on the *reparandum* (the phrase to be repaired) as the antecedent.

Hesitations, another manifestation of disfluency, provide a particularly natural example of self-addressed queries, queries where the intended responder is the original querier:

(92)

a. Carol: Well it's (pause) it's (pause) er (pause) what's his name? Bernard Matthews' turkey roast. (BNC, KBJ)
b. Steve: They're pretty ... um, how can I describe the Finns? They're quite an unusual crowd actually.

Since they can occur at just about any location in a given utterance and their effect is local, disfluencies provide strong motivation for an incremental semantics, that is, a semantics calculated on a word-by-word, left to right fashion (see e.g. Steedman (1999); Kempson *et al.* (2000), and et al (this volume)). Moreover, they require the content construction process to be non-monotonic, since initial decisions can be overriden as a result of self-repair.

(Ginzburg *et al.* (2014b)) sketch how, given an incremental dialogue semantics, accommodating disfluencies is a straightforward extension of the account discussed in section 6 for clarification interaction: the monitoring and update/clarification cycle is modified to happen at the end of each word utterance event, and in case of the need for repair, a repair question gets accommodated into QUD. Self–corrections are handled by a slight generalisation of the rule (66), which just as with the rule QSPEC, underspecifies turn ownership. Hesitations are handled by a CCUR that triggers the accommodation of a question about the identity of the next utterance. Overt examples for such accommodation is exemplified in (92).

9 Conclusions and future directions

In this paper we have presented a theory which encompasses both the analysis of dialogue structure and the traditional concerns of formal semantics. Our main claim is that the two should not be separated. We have used a rich type theory (TTR – type theory with records) in order to achieve this. The main advantages of TTR is that it presents a theory of types which are structured in a similar way to feature structures as employed in feature-based approaches to grammar while at the same time being a type theory including a theory of functions corresponding to the λ-calculus which can be used for a highly intensional theory of semantic interpretation. This type theory can be used to formulate both compositional semantics and the theory of dialogue structure embodied by KoS (Ginzburg, 2012). Among other things we have shown how these tools can be used to create a theory of events (both non-linguistic and linguistic) and thereby create a theory of grammar grounded in the perception of speech events. We have shown how these tools enable us to give an account of the kind of abstract entities needed for semantic analysis, such as propositions and questions. We have also shown how the same tools can be used to given an account of dialogue gameboards and dialogic interaction.

We have exemplified that with respect to variety of phenomena one needs to tackle in order to provide even a rudimentary analysis of an extract from an actual British National Corpus, example (1), which we presented at the beginning of the paper. While we cannot claim to have handled all the details of this example we have nevertheless presented a theory which begins to provide some of the pieces of the puzzle. In particular: *non sentential utterances* are analyzed using a dialogue game-board driven context exemplified in sections 5 and 8.1. *Disfluencies* are handled using conversation rules of a similar form to Clarification Requests and, more generally, to general conversational rules. The possibility of *answering one's own question* is a consequence of factoring turn taking away from illocutionary specification, as in the conversational rule QSPEC. *Misunderstanding* is accommodated by (i) associating different dialogue gameboards with the conversational participants, and (ii) characterizing the grounding and clarification conditions of utterances using locutionary propositions (propositions constructed from utterance types/tokens). *Multilogue* involves scaling up of two-person conversational rules to include communal grounding and acceptance, and multi-agent turn taking. (See Ginzburg & Fernández (2005); Ginzburg (2012))

Beyond the treatment of real conversational interaction, we have looked at a couple of traditional concerns of formal semantics from a dialogical perspective: negation and generalized quantification.

Some other areas which are currently being examined using these tools, but which we have not discussed in this article are: quotation (Ginzburg & Cooper, 2014)—where we argue for the use of utterance types and locutionary propositions as denotations for quotative constructions; the semantics for spatial descriptions and its relationship to robot perception and learning (Dobnik

et al., 2011, 2012; Dobnik & Cooper, 2013); grounding semantics in terms of classifiers used for perception (Larsson, 2013); probabilistic semantics (Cooper *et al.*, 2014); and language acquisition (Larsson & Cooper, 2009; Ginzburg & Moradlou, 2013).

References

et al, Ruth Kempson (this volume), Ellipsis.

Allen, James F., Lenhart K. Schubert, George Ferguson, Peter Heeman, Chung Hee Hwang, Tsuneaki Kato, Marc Light, Nathaniel G. Martin, Bradford W. Miller, Massimo Poesio, & David R. Traum (1995), The trains project: A case study in building a conversational planning agent, *Journal of Experimental and Theoretical AI* 7:7–48.

Artstein, Ron, Mark Core, David DeVault, Kallirroi Georgila, Elsi Kaiser, & Amanda Stent (eds.) (2011), *SemDial 2011 (Los Angelogue): Proceedings of the 15th Workshop on the Semantics and Pragmatics of Dialogue*.

Austin, John L. (1961), Truth, in James Urmson & Geoffrey J. Warnock (eds.), *Philosophical Papers*, Oxford University Press, paper originally published in 1950.

Barwise, Jon & Robin Cooper (1981), Generalized quantifiers and natural language, *Linguistics and Philosophy* 4(2):159–219.

Barwise, Jon & John Etchemendy (1987), *The Liar*, Oxford University Press, New York.

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

Breitholtz, Ellen (2010), Clarification requests as enthymeme elicitors, in *Aspects of Semantics and Pragmatics of Dialogue. SemDial 2010, 14th Workshop on the Semantics and Pragmatics of Dialogue ,*.

Breitholtz, Ellen & Robin Cooper (2011), Enthymemes as rhetorical resources, in Artstein *et al.* (2011).

Brennan, Susan E. & Michael F. Schober (2001), How listeners compensate for disfluencies in spontaneous speech, *Journal of Memory and Language* 44:274–296.

Brown-Schmidt, S., C. Gunlogson, & M.K. Tanenhaus (2008), Addressees distinguish shared from private information when interpreting questions during interactive conversation, *Cognition* 107(3):1122–1134.

Carlson, Lauri (1983), *Dialogue Games*, Synthese Language Library, D. Reidel, Dordrecht.

Clark, Herb & Jean Fox Tree (2002), Using uh and um in spontaneous speech, *Cognition* 84:73–111.

Clark, Herbert (1996), *Using Language*, Cambridge University Press, Cambridge.

Clark, Herbert H & Deanna Wilkes-Gibbs (1986), Referring as a collaborative process, *Cognition* 22(1):1–39.

Clark, H.H. & E.F. Schaefer (1989), Contributing to discourse, *Cognitive science* 13(2):259–294.

Cooper, Robin (2005a), Austinian truth, attitudes and type theory, *Research on Language and Computation* 3:333–362.

Cooper, Robin (2005b), Austinian truth, attitudes and type theory, *Research on Language and Computation* 3(4):333–362.

Cooper, Robin (2005c), Records and record types in semantic theory, *Journal of Logic and Computation* 15(2):99–112.

Cooper, Robin (2010), Generalized quantifiers and clarification content, in Łupkowski & Purver (2010).

Cooper, Robin (2012), Type theory and semantics in flux, in Ruth Kempson, Nicholas Asher, & Tim Fernando (eds.), *Handbook of the Philosophy of Science*, Elsevier BV, volume 14: Philosophy of Linguistics, (271–323), general editors: Dov M. Gabbay, Paul Thagard and John Woods.

Cooper, Robin (2013), Clarification and Generalized Quantifiers, *Dialogue and Discourse* 4(1):1–25.

Cooper, Robin (in prep), Type theory and language: from perception to linguistic communication, draft of book chapters available from `https://sites.google.com/site/typetheorywithrecords/drafts`.

Cooper, Robin, Simon Dobnik, Shalom Lappin, & Staffan Larsson (2014), A probabilistic rich type theory for semantic interpretation, in *Proceedings of the first EACL workshop on Natural Language Semantics and Type Theory*, Gothenburg, (72–79).

Cooper, Robin & Jonathan Ginzburg (2011a), Negation in dialogue, in Artstein *et al.* (2011).

Cooper, Robin & Jonathan Ginzburg (2011b), Negative inquisitiveness and alternatives-based negation, in *Proceedings of the Amsterdam Colloquium, 2011*.

Coquand, Thierry, Randy Pollack, & Makoto Takeyama (2004), A logical framework with dependently typed records, *Fundamenta Informaticae* XX:1–22.

Dobnik, Simon & Robin Cooper (2013), Spatial descriptions in type theory with records, in *Proceedings of IWCS 2013 Workshop on Computational Models of Spatial Language Interpretation and Generation (CoSLI-3)*, Association for Computational Linguistics, Potsdam, Germany, (1–6).

Dobnik, Simon, Robin Cooper, & Staffan Larsson (2012), Modelling language, action and perception in type theory with records, in Denys Duchier & Yannick Parmentier (eds.), *Proceedings of the 7th International Workshop on Constraint Solving and Language Processing (CSLP'12)*, Laboratory for Fundamental Computer Science (LIFO), University of Orléans,, Orléans, France, (51–62).

Dobnik, Simon, Staffan Larsson, & Robin Cooper (2011), Toward perceptually grounded formal semantics, in *Workshop on Integrating Language and Vision on 16 December 2011 at NIPS 2011 (Neural Information Processing Systems)*.

Farkas, Donka & Floris Roelofsen (ms), Polarity particles in an inquisitive discourse model, Manuscript, University of California at Santa Cruz and ILLC, University of Amsterdam.

Fernández, Raquel (2006), *Non-Sentential Utterances in Dialogue: Classification, Resolution and Use*, Ph.D. thesis, King's College, London.

Fernando, Tim (2004), A finite-state approach to events in natural language semantics, *Journal of Logic and Computation* 14(1):79–92.

Fernando, Tim (2006), Situations as strings, *Electronic Notes in Theoretical Computer Science* 165:23–36.

Fernando, Tim (2008), Finite-state descriptions for temporal semantics, in Harry Bunt & Reinhart Muskens (eds.), *Computing Meaning, Volume 3*, Springer, volume 83 of *Studies in Linguistics and Philosophy*, (347–368).

Fernando, Tim (2009), Situations in LTL as strings, *Information and Computation* 207(10):980–999, ISSN 0890-5401, doi:DOI:10.1016/j.ic.2008.11.003.

Fillmore, Charles J. (1985), Frames and the semantics of understanding, *Quaderni di Semantica* 6(2):222–254.

Gibson, James J. (1986), *The Ecological Approach to Visual Perception*, Lawrence Erlbaum Associates.

Ginzburg, Jonathan (1995), Resolving questions, I, *Linguistics and Philosophy* 18:459–527.

Ginzburg, Jonathan (1997), On some semantic consequences of turn taking, in P. Dekker, M. Stokhof, & Y. Venema (eds.), *Proceedings of the 11th Amsterdam Colloquium on Formal Semantics and Logic*, ILLC, Amsterdam, (145–150).

Ginzburg, Jonathan (2011), Situation semantics and the ontology of natural language, in Klaus von Heusinger, Claudia Maierborn, & Paul Portner (eds.), *The Handbook of Semantics*, Walter de Gruyter.

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

Ginzburg, Jonathan & Robin Cooper (2014), Quotation via dialogical interaction, *Journal of Logic, Language, and Information* 23(3) 287–311.

Ginzburg, Jonathan, Robin Cooper, & Tim Fernando (2014a), Propositions, questions, and adjectives: a rich type theoretic approach, in *Proceedings of the first EACL workshop on Natural Language Semantics and Type Theory*, Gothenburg.

Ginzburg, Jonathan & Raquel Fernández (2005), Scaling up to multilogue: some benchmarks and principles, in *Proceedings of the 43rd Meeting of the Association for Computational Linguistics*, Michigan, (231–238).

Ginzburg, Jonathan, Raquel Fernández, & David Schlangen (2014b), Disfluencies as intra-utterance dialogue moves, *Semantics and Pragmatics* 7(9) 1–64.

Ginzburg, Jonathan & Sara Moradlou (2013), The earliest utterances in dialogue: towards a formal theory of parent/child talk in interaction, in Raquel Fernández & Amy Isard (eds.), *Proceedings of SemDial 2013 (DialDam)*, University of Amsterdam.

Ginzburg, Jonathan & Matt Purver (2012), Quantfication, the reprise content hypothesis, and type theory, in Staffan Larsson & Lars Borin (eds.), *From Quantfication to Conversation: Festschrift for Robin Cooper on the occasion of his 65th birthday*, College Publications, volume 19 of *Tributes*.

Ginzburg, Jonathan & Ivan A. Sag (2000), *Interrogative Investigations: the form, meaning and use of English Interrogatives*, number 123 in CSLI Lecture Notes, CSLI Publications, Stanford: California.

Godfrey, John J., E. C. Holliman, & J. McDaniel (1992), Switchboard: Telephone speech corpus for research and devlopment, in *Proceedings of the IEEE Conference on Acoustics, Speech, and Signal Processing*, San Francisco, USA, (517–520).

Groenendijk, Jeroen & Martin Stokhof (1997), Questions, in Johan van Benthem & Alice ter Meulen (eds.), *Handbook of Logic and Linguistics*, North Holland, Amsterdam.

Hamblin, C. L. (1973), Questions in Montague English, in Barbara Partee (ed.), *Montague Grammar*, Academic Press, New York.

Healey, P.G.T., M. Purver, J. King, J. Ginzburg, & G. Mills (2003), Experimenting with clarification in dialogue, in R. Alterman & D. Kirsh (eds.), *Proceedings of the 25th Annual Conference of the Cognitive Science Society*, Mahwah, N.J.: LEA, (539–544.).

Heeman, Peter A. & James F. Allen (1999), Speech repairs, intonational phrases and discourse markers: Modeling speakers' utterances in spoken dialogue, *Computational Linguistics* 25(4):527–571.

Hoepelmann, Jacob (1983), On questions, in Ferenc Kiefer (ed.), *Questions and Answers*, Reidel.

Hopcroft, John E. & Jeffrey D. Ullman (1979), *Introduction to Automata Theory, Languages and Computation*, Addison-Wesley Publishing, Reading Massachusetts.

Kempson, Ruth, Wilfried Meyer-Viol, & Dov Gabbay (2000), *Dynamic Syntax: The Flow of Language Understanding*, Blackwell, Oxford.

Larsson, Staffan (2013), Formal semantics for perceptual classification, *Journal of Logic and Computation* doi:10.1093/logcom/ext059.

Larsson, Staffan & Robin Cooper (2009), Towards a formal view of corrective feedback, in *Proceedings of the EACL 2009 Workshop on Cognitive Aspects of Computational Language Acquisition*, Athens.

Levelt, Willem J. (1983), Monitoring and self-repair in speech, *Cognition* 14(4):41–104.

Luo, Zhaohui (2011), Contextual Analysis of Word Meanings in Type-Theoretical Semantics, in Sylvain Pogodalla & Jean-Philippe Prost (eds.), *Logical Aspects of Computational Linguistics: 6th International Conference, LACL 2011*, Springer, number 6736 in Lecture Notes in Artificial Intelligence, (159–174).

Łupkowski, Paweł & Jonathan Ginzburg (2014), Question answers, *Computational Linguistics (Under Review)* .

Łupkowski, Paweł & Matthew Purver (eds.) (2010), *Aspects of Semantics and Pragmatics of Dialogue. SemDial 2010, 14th Workshop on the Semantics and Pragmatics of Dialogue*, Polish Society for Cognitive Science, Poznań.

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

Michaelis, Laura A. (2009), Sign-based construction grammar, in *The Oxford Handbook of Linguistic Analysis*, Oxford University Press.

Montague, Richard (1973), The Proper Treatment of Quantification in Ordinary English, in Jaakko Hintikka, Julius Moravcsik, & Patrick Suppes (eds.), *Approaches to Natural Language: Proceedings of the 1970 Stanford Workshop on Grammar and Semantics*, D. Reidel Publishing Company, Dordrecht, (247–270).

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

Partee, B.H., A.G.B. ter Meulen, & R.E. Wall (1990), *Mathematical Methods in Linguistics*, Springer.

Peters, Stanley & Dag Westerståhl (2006), *Quantifiers in Language and Logics*, Oxford University Press.

Purver, M. (2006), Clarie: Handling clarification requests in a dialogue system, *Research on Language & Computation* 4(2):259–288.

Purver, Matt & Jonathan Ginzburg (2004), Clarifying noun phrase semantics, *Journal of Semantics* 21(3):283–339.

Purver, Matthew, Jonathan Ginzburg, & Patrick Healey (2006), Lexical categories and clarificational potential, revised version under review.

Purver, Matthew, Eleni Gregoromichelaki, Wilfried Meyer-Viol, & Ronnie Cann (2010), Splitting the *I*s and Crossing the *You*s: Context, Speech Acts and Grammar, in Łupkowski & Purver (2010), (43–50).

Purver, Matthew, Patrick G. T. Healey, James King, Jonathan Ginzburg, & Greg J. Mills (2003), Answering clarification questions, in *Proceedings of the 4th SIGdial Workshop on Discourse and Dialogue*, ACL, Sapporo.

Ranta, Aarne (this volume), Intuitionistic type theory and dependent types.

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

Schegloff, Emanuel (2007), *Sequence Organization in Interaction*, Cambridge University Press, Cambridge.

Schegloff, Emanuel, Gail Jefferson, & Harvey Sacks (1977), The preference for self-correction in the organization of repair in conversation, *Language* 53:361–382.

Searle, John R. (1969), *Speech Acts: an Essay in the Philosophy of Language*, Cambridge University Press.

Shieber, Stuart (1986), *An Introduction to Unification-Based Approaches to Grammar*, CSLI Publications, Stanford.

Shriberg, Elizabeth E. (1994), *Preliminaries to a theory of speech disfluencies*, Ph.D. thesis, University of California at Berkeley, Berkeley, USA.

Steedman, Mark (1999), *The Syntactic Process*, Linguistic Inquiry Monographs, MIT Press, Cambridge.

Wiśniewski, Andrzej (2001), Questions and inferences, *Logique et Analyse* 173:5–43.

Wiśniewski, Andrzej (2003), Erotetic search scenarios, *Synthese* 134:389–427.

Wiśniewski, Andrzej (this volume), The semantics of questions.