A Dynamic Categorial Grammar*

Scott Martin[1] and Carl Pollard[2]

[1] Natural Language Understanding and Artificial Intelligence Laboratory
Nuance Communications
1198 East Arques Avenue
Sunnyvale, California 94085 USA
scott.martin@nuance.com
[2] Department of Linguistics
The Ohio State University
1712 Neil Avenue
Columbus, Ohio 43210 USA
pollard@ling.ohio-state.edu

Abstract. We present a compositional, dynamic categorial grammar for discourse analysis that captures the core insights of dynamic semantics: indefinites do not quantify but introduce discourse referents; definites are anaphoric to previously-mentioned discourse referents; discourse referents have their 'lifespan' limited by certain operators. The categorial grammar formalism we propose is strongly lexicalist and derives linguistic signs with a syntactic division of labor separating surface form from the underlying combinatorics. We argue that this formalism compares favorably with earlier efforts on several counts. It does not require any complicated or idiosyncratic machinery such as specialized assignments, states, or continuations, and encodes the requirement that a certain discourse referent be present in the discourse context using dependent types, rather than e.g. partial functions. The dynamic semantics itself is a straightforward extension of an underlying static semantics that is fully (hyper)intensional, avoiding many unsavory problems associated with standard possible worlds approaches.

Keywords: categorial grammar, dynamic semantics, compositionality, dependent type theory, hyperintensionality

1 Introduction

In dynamic semantics, the interpretation of sentences both depends upon and acts upon the utterance context. When the classic dynamic semantic frameworks of discourse representation theory (DRT, [15]) and file change semantics (FCS, [13]) were first introduced, they attracted a lot of attention because they provided analyses of phenomena (such as donkey anaphora, cross-sentential anaphora,

* Thanks are due to audiences at Ohio State University who critiqued early drafts of this work, especially to the joint linguistics/philosophy seminar in the spring of 2011 and the pragmatics reading group.

presupposition satisfaction, and the novelty condition on indefinites) that the then-predominant semantic framework of Montague semantics (MS, [24]) could not account for. However, in formal terms, classic dynamic semantics compared unfavorably with MS, which provided an explicit (albeit awkward) compositional interface to syntax and whose semantic theory was expressed in a higher-order language essentially equivalent to the classical simple theory of types ([3, 6, 14]). Neither DRT nor FCS was compositional; the theoretical content of FCS was expressed (not always as precisely as might have been hoped for) only in the metalanguage, and DRT struck many as excessively syntactic and procedural.

Early attempts at logical formalization of dynamic semantics such as dynamic predicate logic (DMG, [8]) and dynamic Montague grammar (DPL, [9]) addressed the compositionality issue, but suffered from idiosyncratic, inelegant handling of variables (failure of alphabetic variance, destructive assignments, scope extension of existential quantifiers, etc.). Muskens [25] seems to have been the first to demonstrate that the fundamental insights of dynamic semantics could be captured in a compositional way without going beyond the expressive limits of classical type theory. But his approach incorporated some nonstandard—and unnecessary—features, such as an additional basic type for *states* and an explicit encoding of DRT accessibility conditions. More recent type-theoretic embodiments of dynamic semantics [1, 4, 11], including our earlier efforts [19, 20, 22, 23] are free of these particular defects, but room for improvement remains.

In this paper, we propose a new framework for compositional type-theoretic dynamic semantics which improves on previous proposals in the following respects: (1) It comes equipped with a straightforward interface to a linear-logic-based categorial grammar (LCG) along the general lines of [10, 26, 27], etc. (2) Although grammars and the derivations they generate make no reference to possible worlds (or extensions of meanings at them), the underlying semantic theory is fully (hyper)intensional from the get-go, so that it can be straightforwardly extended to handle propositional attitudes, evidentiality [17], supplements [21], interrogatives [31], etc. (3) Since it permits, but does not require, the modeling of (static) propositions as sets of possible worlds, one can choose between the familiarity of (intensional) MS and a weaker, hyperintensional, static underpinning [28–30] that avoids MS's notorious foundational problems (e.g. the granularity problem). (4) It straightforwardly models contexts as functions from tuples of entities ('discourse referents', abbreviated 'DRs') to propositions ('common ground', abbreviated 'CG'), and updates as functions from contexts to contexts, as in FCS, obviating the need for states [25] or continuations [11]. (5) Following the original insights of both DRT and FCS, the semantics of indefinites ('dynamic existential quantification') is not defined in terms of static existential quantification, so there is no need for any notion of scope extension. (6) Updates are explicitly bifurcated into the 'carryover' from the input context and the new content proffered by the current utterance, thereby providing a hook for an envisioned extension covering utterance acceptance and rejection. (7) There is no

requirement that common grounds remain consistent; instead, and more realistically, *perceived* inconsistency could be grounds for rejecting proffered content.

2 Syntactic Framework

In place of Montague's categorial grammar, we use a linear-logic-based form of categorial grammar (hereafter, LCG) broadly similar to de Groote's ACG [10] or Muskens's λ-grammar [26]. An LCG generates tripartite **signs**, such as:

$$\vdash \text{pedro} \,;\, \text{NP} \,;\, \mathsf{p} : \text{e}$$
$$\vdash \text{donkey} \,;\, \text{N} \,;\, \mathsf{donkey} : \text{p}_1$$

The three components of a sign are called, respectively, the **phenogrammatical** component (**pheno** for short), the **tectogrammatical** component (**tecto** for short), and the **semantic** component (**semantics** for short). (The semantic types will be explained in due course.) In the tecto, there is only one connective $\multimap$ (linear implication) instead of directional slashes / and \.

The pheno component of a sign need not be a string (basic type s), but can be a (possibly higher order) function over strings that tells how a functor is to be ordered with respect to its arguments:

$$\vdash \lambda_s.s \cdot \text{brays} \,;\, \text{NP} \multimap \text{S} \,;\, \mathsf{bray} : \text{p}_1$$
$$\vdash \lambda_{st}.s \cdot \text{beats} \cdot t \,;\, \text{NP} \multimap \text{NP} \multimap \text{S} \,;\, \mathsf{beat} : \text{p}_2$$
$$\vdash \lambda_{sf}.f\ (\text{every} \cdot s) \,;\, \text{N} \multimap (\text{NP} \multimap \text{S}) \multimap \text{S} \,;\, \mathsf{every} : \text{dt}$$

Here $s, t : \text{s}$ and $f : \text{s} \to \text{s}$. The higher-order pheno theory axiomatizes strings to form a monoid with identity $\mathbf{e} : \text{s}$ (null string) and associative operation $\cdot : \text{s} \to \text{s} \to \text{s}$ (concatenation, written infix). All other pheno constants employed, representing word phonologies, are of type s, including $\mathbf{e}$ (null string).

Besides lexical entries, which serve as nonlogical axioms, there is a schema of logical axioms

$$p : P \,;\, A \,;\, z : B \vdash p : P \,;\, A \,;\, z : B \quad ,$$

instances of which correspond to traces in mainstream generative grammar (MGG), or to Montague's 'syntactic variables'. That is: a trace is a hypothetical sign, with variables for pheno and semantics. Here the type metavariables P, A, and B range, respectively, over **pheno types** (s and implicative types over s), **tecto types** (basic tecto types such as N, NP and S, and linear implicative types over them), and **sense types** (to be defined in due course).

For example, an NP trace looks like this:

$$t : \text{s} \,;\, \text{NP} \,;\, x : \text{e} \vdash t : \text{s} \,;\, \text{NP} \,;\, x : \text{e}$$

As in the implicative fragment of intuitionistic linear propositional logic, the only tectogrammatical rules are **modus ponens**:

$$\frac{\Gamma \vdash f : A \to D \,;\, B \multimap E \,;\, g : C \to F \qquad \Delta \vdash a : A \,;\, B \,;\, c : C}{\Gamma, \Delta \vdash f\ a : D \,;\, E \,;\, g\ c : F} \,,$$

and **hypothetical proof**:

$$\frac{\Gamma, p : P\,;\, A\,;\, z : B \vdash a : C\,;\, D\,;\, b : E}{\Gamma \vdash \lambda_p.a : P \to C\,;\, A \multimap D\,;\, \lambda_z.b : B \to E}\,,$$

which are, roughly, LCG's counterparts of MGG's Merge and (the trace-binding aspect of) Move. These correspond, respectively, to application and abstraction in the pheno and semantic components.

Finally, the LCG counterparts of Montague's analysis trees are (pheno- and semantics-labeled) sequent-style linear-logic proof trees whose leaves are axioms (i.e. either lexical entries or traces). The following three proofs use only modus ponens:

$$\frac{\vdash \lambda_s.s \cdot \text{brayed}\,;\, \text{NP} \multimap \text{S}\,;\, \mathsf{bray} \qquad \vdash \text{chiqita}\,;\, \text{NP}\,;\, \mathsf{c}}{\vdash \text{chiqita} \cdot \text{brayed}\,;\, \text{S}\,;\, \mathsf{bray\ c}}$$

$$\frac{\dfrac{\vdash \lambda_{st}.s \cdot \text{beats} \cdot t\,;\, \text{NP} \multimap \text{NP} \multimap \text{S}\,;\, \mathsf{beat} \qquad \vdash \text{pedro}\,;\, \text{NP}\,;\, \mathsf{p}}{\vdash \lambda_t.\text{pedro} \cdot \text{beats} \cdot t\,;\, \text{NP} \multimap \text{S}\,;\, \mathsf{beat\ p}} \qquad \vdash \text{chiqita}\,;\, \text{NP}\,;\, \mathsf{c}}{\vdash \text{pedro} \cdot \text{beats} \cdot \text{chiquita}\,;\, \text{S}\,;\, \mathsf{beat\ p\ c}}$$

$$\frac{\dfrac{\vdash \lambda_{sf}.f\ (\text{every} \cdot s)\,;\, \text{Det}\,;\, \mathsf{every} \qquad \vdash \text{donkey}\,;\, \text{N}\,;\, \mathsf{donkey}}{\vdash \lambda_f.f\ \text{every} \cdot \text{donkey}\,;\, \text{QP}\,;\, \mathsf{every\ donkey}} \qquad \vdash \lambda_s.s \cdot \text{brays}\,;\, \text{NP} \multimap \text{S}\,;\, \mathsf{bray}}{\vdash \text{every} \cdot \text{donkey} \cdot \text{brays}\,;\, \text{S}\,;\, \mathsf{every\ donkey\ bray}}$$

In the last of these, QP abbreviates (NP $\multimap$ S) $\multimap$ S, and Det abbreviates N $\multimap$ QP. And the following proof uses a trace and an instance of hypothetical proof to 'lower' *every donkey* into the object position. Here we abbreviate subtrees by numerical labels:

(1)
$$\frac{\vdash \lambda_{sf}.f\ (\text{every} \cdot s)\,;\, \text{Det}\,;\, \mathsf{every} \qquad \vdash \text{donkey}\,;\, \text{N}\,;\, \mathsf{donkey}}{\vdash \lambda_f.f\ \text{every} \cdot \text{donkey}\,;\, \text{QP}\,;\, \mathsf{every\ donkey}}$$

(2)
$$\frac{\dfrac{\dfrac{\vdash \lambda_{st}.s \cdot \text{beats} \cdot t\,;\, \text{NP} \multimap \text{NP} \multimap \text{S}\,;\, \mathsf{beat} \qquad \vdash \text{pedro}\,;\, \text{NP}\,;\, \mathsf{p}}{\vdash \lambda_t.\text{pedro} \cdot \text{beats} \cdot t\,;\, \text{NP} \multimap \text{S}\,;\, \mathsf{beat\ p}} \qquad s\,;\, \text{NP}\,;\, x \vdash s\,;\, \text{NP}\,;\, x}{s\,;\, \text{NP}\,;\, x \vdash \text{pedro} \cdot \text{beats} \cdot s\,;\, \text{S}\,;\, \mathsf{beat\ p}\ x}}{\vdash \lambda_s.\text{pedro} \cdot \text{beats} \cdot s\,;\, \text{NP} \multimap \text{S}\,;\, \lambda_x.\mathsf{beat\ p}\ x}$$

$$\frac{(1) \qquad\qquad (2)}{\vdash \text{pedro} \cdot \text{beats} \cdot \text{every} \cdot \text{donkey}\,;\, \text{S}\,;\, \mathsf{every\ donkey}\ (\lambda_x.\mathsf{beat\ p}\ x)}$$

This technology, due to Oehrle [27], plays the same role in LCG that quantifier lowering does in Montague grammar.

3 The Underlying Logic and Notational Conventions

Our semantic theory is couched in a classical higher order logic (HOL) with entities (e), propositions (p, cf. [32]), and natural numbers (n) as basic types, in addition to truth values (t) courtesy of the logic itself. Unlike MS, where propositions are sets of worlds and so the boolean structure on them is parasitic on that of the truth values, we axiomatize (in the following section) that propositions (in the sense of static declarative sentence meanings) form a preboolean algebra relative to (propositional) entailment. Besides the usual cartesian type constructors $\to$ (exponential), $\times$ (product), and T (unit type), we also make use of dependent products ($\Pi_{n:\mathrm{n}}.A$) and sums ($\Sigma_{n:\mathrm{n}}.A$) that depend on a natural number.

We adopt the following notational conventions. For any type A, the n-fold cartesian product is written A^n; hence $A^0 = \mathrm{T}$, $A^1 = A$, and $A^{n+1} = A^n \times A$ ($n > 0$). Implication associates to the right, so $A \to B \to C$ abbreviates $A \to (B \to C)$, and similarly for the linear implication $\multimap$ used in the tectogrammar. Outermost parentheses of terms are often deleted. Application associates to the left, so $f\ b\ a$ abbreviates $(f\ b)\ a$. For terms whose type is a cartesian power A^n ($n > 0$), we often write the vector notation $\mathbf{x}$ in place of $(x_0, \ldots, x_{n-1})$. The first occurrence of a 'vector' variable $\mathbf{x}$ is often mnemonically superscripted with its number of components, thus $\mathbf{x}^n$. Abstraction on product types is written either $\lambda_{\mathbf{x}}.a$ or $\lambda_{x_0,\ldots,x_{n-1}}.a$, while abstraction on multiple variables *without* commas abbreviates *successive* abstraction, e.g. $\lambda_{xy}.a$ abbreviates $\lambda_x.\lambda_y.a$. If $x : A$, $N : B$, and $M : (A \to B) \to C$, we abbreviate $M\ \lambda_x.N$ to $M_x.N$.

4 The Underlying Static Semantic Theory

4.1 Static semantic types

The underlying static semantic theory is **agnostic hyperintensional semantics (AHS, [28, 30])**, a possible worlds semantics with fine-grained meanings. 'Agnostic' here means that the theory is indifferent as to whether propositions are sets of worlds (as in MS) or worlds are (in one-to-one correspondence with) maximal consistent sets of propositions (as in the 'first-wave' possible-worlds theories of Wittgenstein and C.I. Lewis), though the latter is by far our personal preference. (In fact, AHS is *logically weaker* than Montague semantics (MS): adding one axiom turns it into MS.)

We assume the basic types e (entities), p ((static) propositions), and w (worlds). (The type w is never mentioned in the grammar or in analyses of expressions; it only comes up when using the semantic theory to reason about extensions.) It's also convenient to define (for $n \geq 0$) the types of n-ary *static properties* by:

$$\mathrm{p}_0 =_{\mathrm{def}} \mathrm{p}$$
$$\mathrm{p}_{n+1} =_{\mathrm{def}} \mathrm{e} \to \mathrm{p}_n$$

For example, the types for *static generalized quantifiers (over entities)* and *static generalized determiners (over entities)* are, respectively, of types:

$$\mathrm{q} =_{\mathrm{def}} \mathrm{p}_1 \to \mathrm{p}$$
$$\mathrm{dt} =_{\mathrm{def}} \mathrm{p}_1 \to \mathrm{p}_1 \to \mathrm{p}$$

4.2 Static semantic constants

Side-by-side with the usual truth-value connectives and quantifiers of the underlying HOL (true, false, $\wedge$, $\vee$, $\to$, $\neg$, $\exists$, and $\forall$), there are the propositional connectives truth, $\mathsf{falsity}$, and, or, $\mathsf{implies}$, and not, and the propositional quantifiers exists_A, and forall_A. The latter are polymorphic over sense types (see following subsection) A. Some determiner meanings can be defined in terms of these (omitting the type subscripts):

$$\mathsf{every} =_{\mathrm{def}} \lambda_{PQ}.\mathsf{forall}_x.(P\ x)\ \mathsf{implies}\ (Q\ x)$$
$$\mathsf{a} =_{\mathrm{def}} \lambda_{PQ}.\mathsf{exists}_x.(P\ x)\ \mathsf{and}\ (Q\ x)$$

These connectives and quantifiers are subject to meaning postulates that relate them to their truth-value counterparts in the expected way (see [28] or [30] for details). As a consequence, AHS is *finer-grained* than MS: senses which agree in extension at every world need not be identical. For example, not all inconsistent CGs are interchangeable, even though they all entail every proposition. What matters from the dynamic perspective is whether a discourse participants (DPs) can *detect* that a CG would be rendered inconsistent by conjoining to it the proffered content of the current utterance (which would constitute grounds for declining to admit it to the CG).

4.3 Static meanings and their extensions

Types to which static meanings can belong are called **sense types**. These are T, e, p, and function types formed from these. Each sense type A has a corresponding **extension** type $\mathrm{Ext}(A)$ defined as follows:

$$\mathrm{Ext}(\mathrm{T}) = \mathrm{T}$$
$$\mathrm{Ext}(\mathrm{e}) = \mathrm{e}$$
$$\mathrm{Ext}(\mathrm{p}) = \mathrm{t} \qquad \text{(following Frege [5])}$$
$$\mathrm{Ext}(A \to B) = A \to \mathrm{Ext}(B)$$

For each sense type A, $A \to \mathrm{p}$ is called the type of A-**properties**, and $A \to \mathrm{t}$ is called the type of A-**sets**. If w is a world and $a : A$ a sense, then the **extension** of a at w is written $a\ @\ w$. (So @ is really a family of constants $@_A : A \to \mathrm{w} \to \mathrm{Ext}(A)$, written infix.) Following, roughly, Kripke [18], we assume every entity is its own extension at every world:

$$\vdash \forall_{x:\mathrm{e}}.\forall_{w:\mathrm{w}}.x\ @\ w = x$$

For any proposition p, $p \mathbin{@} w$ is called, following [5], the **truth value** of p at w. The extensions of senses with a functional type $A \to B$ are given by

$$\vdash \forall_{f:A\to B}.\forall_{w:\mathrm{w}}.f \mathbin{@} w = \lambda_x.(f\ x) \mathbin{@} w\ .$$

In particular,

$$\vdash \forall_{P:\mathrm{p}_1}.\forall_{w:\mathrm{w}}.P \mathbin{@} w = \lambda_x.(P\ x) \mathbin{@} w\ .$$

For example, if **w** is a world, then the extension at **w** of the donkey property is the set of all entities a such that the proposition that a is a donkey is true at **w**.

5 Hyperintensional Dynamic Semantics

5.1 Contexts

To model contexts, we begin with our counterparts to FCS's *assignments*, namely *tuples* of entities. The linear positions in these tuples play the 'addressing' role played by DRs (in DRT) or file cards (in FCS). What about contexts? On the Stalnaker/Lewis conception, this is supposed to be the set of propositions in the CG, or else a single proposition which is the conjunction of those. We modify that view to treat the CG not as a proposition (type p), but rather as *a function from tuples of entities to propositions* (type $\mathrm{e}^n \to \mathrm{p}$), where n is the number of 'live' DRs.[3]. The philosophy behind this typing is that the DPs don't really have a proposition in common, since in general the identity of the DRs that the discourse is about (typically, introduced into the discourse by uses of indefinite NPs) are not known. Rather, what they have in common is only a function from n-tuples of entities to propositions, which would give rise to a proposition in the obvious way if only the identities of the entities were known. To put it another way: the DRs are 'identified' only in terms of what the CG says about them. To give a highly simplified example, suppose (counterfactually!) that there are actual 'out-of-the-blue' utterances where the input CG is empty.[4] Then the output context from an out-of-the-blue utterance of *a farmer beat a donkey* will be

$$\lambda_{x,y}.(\mathsf{farmer}\ x)\ \mathsf{and}\ (\mathsf{donkey}\ y)\ \mathsf{and}\ (\mathsf{beat}\ x\ y)\ ,$$

or, using the equivalent vector notation,

$$\lambda_{\mathbf{x}^2}.(\mathsf{farmer}\ x_0)\ \mathsf{and}\ (\mathsf{donkey}\ x_1)\ \mathsf{and}\ (\mathsf{beat}\ x_0\ x_1)\ .$$

Note that this is just an uncurried binary static property. In particular there is no existential quantification. This reflects the fundamental insight of many versions

[3] However, we continue to use the type p for CGs where $n = 0$, in preference to the mathematically more elegant but notationally awkward $\mathrm{T} \to \mathrm{p}$.

[4] Technically, this is modeled by truth, the designated top element in the preboolean algebra of static propositions.

of dynamic semantics that indefinites (and also definites) are *not* quantificational in nature. (However, as we will see, they have the same dynamic semantic type as 'truly quantificational' NPs.) Intuitively, the context corresponds not to an existential proposition, but to mutual acceptance that *whichever farmer it is that we're talking about* beats *whichever donkey it is that we're talking about.* (We could turn this uncurried property into a proposition by applying the static existential quantifier $\mathsf{exists}_{\mathrm{e}\times\mathrm{e}}$ to it, and then define the context to be 'true' in any world where that proposition is true.)

Based on these considerations, we now *define* the type of n-ary **contexts** c_n to be simply p if $n = 0$ and $\mathrm{e}^n \to \mathrm{p}$ if $n > 0$; and the type c of **contexts** is defined to be the dependent sum of all these:

$$\mathrm{c} =_{\mathrm{def}} \Sigma_{n:\mathrm{n}}.\mathrm{c}_n$$

Also, we define the **arity** of an n-ary context c, written $|c|$, to be n. (So, since technically, since a member of an n-indexed sum is an ordered pair of a natural number n and a member of the n-th cofactor, the arity of a context is just its first component.) Intuitively, $|c|$ is the number of DRs that c is about. For example, $|c| = 2$ if c is the context $\lambda_{x,y}.(\mathsf{farmer}\ x)\ \mathsf{and}\ (\mathsf{donkey}\ y)\ \mathsf{and}\ (\mathsf{beat}\ x\ y)$ discussed above. As we'll see, the DRs are potential targets of subsequent anaphora.

In particular, a nullary context (same as a static proposition) is, intuitively, a context where the DPs have nothing in common to talk about. A special case of a nullary context is the **null context**

$$\mathrm{T} =_{\mathrm{def}} \mathsf{truth}\ ,$$

also known as **out of the blue**, where $\mathsf{truth} : \mathrm{p}$ is some obvious necessary truth. This models the content where the DPs have no DRs to talk about given in advance, and not even anything they have agreed to take for granted. Realistically, discourse is never completely out of the blue; even the driver and the hitchhiker can agree that the driver picked the hitchhiker up, and have some DRs to talk about (the weather, them Bucks, the car, the boring scenery, etc.).

When we consider anaphora, it will be important to have a handle on how many DRs the context knows about, because we will analyze anaphoric expressions (e.g. definite pronouns) as essentially n-ways *ambiguous*, where n is the arity of the utterance context. For any natural number n, an anaphoric reference to the n-th DR will only interpretable in a context whose arity is greater than n. The type of such entities, called $\mathrm{c}_{>n}$, is defined as the dependent sum

$$\mathrm{c}_{>n} =_{\mathrm{def}} \mathrm{c}_{\geq n+1}\ ,$$

where

$$\mathrm{c}_{\geq m} =_{\mathrm{def}} \Sigma_{n:\mathrm{n}}.\mathrm{c}_{m+n}\ .$$

5.2 Toward Dynamic Senses

To dynamicize LCG, we have to replace the static senses with dynamic ones. For example, as we'll see, the dynamic counterpart of type e is the type n of

natural numbers (thought of as DRs). And the dynamic counterpart of type p will be the type u of **updates**, which correspond roughly to FCS's context change potentials (CCPs). To a first approximation, updates are functions from contexts to contexts (type c → c), but there is a catch: if an update makes anaphoric reference to something in the context, then it is only defined on contexts that contain a suitable antecedent to which the anaphoric reference can be resolved. We'll use dependent typing to give u a more subtle definition than c → c that solves this problem (and which does not require partial functions, which our type theory does not countenance).

Again, to a first approximation, dynamic counterparts of other sense types are obtained in the expected way from these basic dynamic types. For example, the dynamic counterpart of e → p (properties of entities) is (to a first approximation) the type n → u of *dynamic properties*; and the dynamic counterpart of (e → p) → p (GQs) is (to a first approximation) the type (n → u) → u of *dynamic generalized quantifiers (DGQs)*.

5.3 Contents vs. Updates

In discourse, when an utterance is accepted, the update carries over the common ground of the input context, while conjoining to it new content contributed by the utterance. To capture this fundamental intuition, we distinguish between two different things of the same type: updates, which correspond to FCS's CCPs, and **contents**, the meaning contribution of the new utterance (here, we consider only assertions). For any content k which is accepted by the DPs, the induced update is obtained by applying to it a certain function called cc (mnemonic for 'context change'). As we'll see, cc applies first to a content k and then to an input context c to produce the new context $\mathsf{cc}\ k\ c$ for the next utterance. Thus the update $\mathsf{cc}\ k$ is itself a function which converts an input context into a new context into which the newly accepted content k has been incorporated.

So, naively, it *looks* as though the type u for both contents and updates should be c → c, and the type for cc should be (c → c) → (c → c). But there is a subtlety: different contents (and the updates they induce upon acceptance) have different *degrees*: the number of new DRs that are introduced. As we'll see,

$$|\mathsf{cc}\ k| = |\mathsf{cc}\ k\ c| = |c| + |k| \ ,$$

so the arity of the output context is the arity of the input context plus the degree of the utterance content. (Like the arity of a context, the degree of a content k is written $|k|$.)

Now, for any natural number n, the type of n-degree updates is that of a function that maps a context c to a new context whose degree is $|c| + n$. Accordingly, we define the type of n-**degree updates** to be the dependent product

$$\mathrm{u}_n =_{\mathrm{def}} \Pi_{c:\mathrm{c}}.\mathrm{c}_{|c|+n} \ .$$

Then the type u of **updates** is the dependent sum of the u_n as n ranges over all natural numbers:

$$\mathrm{u} =_{\mathrm{def}} \Sigma_{n:\mathrm{n}}.\mathrm{u}_n = \Sigma_{n:\mathrm{n}}.\Pi_{c:\mathrm{c}}.\mathrm{c}_{|c|+n}$$

The types of contents are defined similarly, as $\mathrm{k}_n =_{\mathrm{def}} \mathrm{u}_n$ and $\mathrm{k} =_{\mathrm{def}} \mathrm{u}$. With these type definitions in place, we can now define the context change function as follows:

$$\mathsf{cc} =_{\mathrm{def}} \lambda_{k:\mathrm{k}}.\lambda_{c:\mathrm{c}}.\lambda_{\mathbf{x}^{|c|},\mathbf{y}^{|k|}}.(c\ \mathbf{x})\ \mathsf{and}\ (k\ c\ \mathbf{x},\mathbf{y})$$

That is: cc takes as arguments a content k with degree $n = |k|$ and a context c with arity $m = |c|$ and returns a new context $\lambda_{\mathbf{x}^m,\mathbf{y}^n}.(c\ \mathbf{x})\ \mathsf{and}\ (k\ c\ \mathbf{x},\mathbf{y})$. This new context is a function that maps any $m+n$ entities to a (static) proposition which is itself the conjunction of two propositions: (1) $(c\ \mathbf{x})$, the 'carryover', which is what the input context had already established about the first m DRs; and (2) $(k\ c\ \mathbf{x},\mathbf{y})$, the new contribution, which is what the current utterance's content says about *all* the DRs (the m original ones plus the n new ones that it introduces) *in* that context. For example, the content of *it's raining* is the 0-degree content

$$\textsc{rain} =_{\mathrm{def}} \lambda_{c:\mathrm{c}}.\lambda_{\mathbf{x}^{|c|}}.\mathsf{rain}\ .$$

So the associated update is

$$\vdash \mathsf{cc}\ \textsc{rain} = \lambda_{c:\mathrm{c}}.\lambda_{\mathbf{x}^{|c|}}.(c\ \mathbf{x})\ \mathsf{and}\ (\textsc{rain}\ c\ \mathbf{x}) = \lambda_{c:\mathrm{c}}.\lambda_{\mathbf{x}^{|c|}}.(c\ \mathbf{x})\ \mathsf{and}\ \mathsf{rain}\ .$$

5.4 From linear categorial grammar to dynamic categorial grammar

Almost all the work involved in dynamicizing an LCG consists of replacing word meanings with their dynamic counterparts, since the logical rules and axioms (traces) of LCG carry over to DyCG unchanged. We will turn to that task presently. However, we also need one new, nonlogical, grammar rule, **continue**, whose purpose is to continue a discourse (tecto type D) by the addition of the next accepted utterance:

$$\frac{\vdash s\ ;\mathrm{D}\ ;u \qquad \vdash t\ ;\mathrm{S}\ ;k}{\vdash s\cdot t\ ;\mathrm{D}\ ;\lambda_{c:\mathrm{c}}.\mathsf{cc}\ k\ (u\ c)}$$

Note that the sequent contexts are empty: binding of traces, and therefore '*wh*-movement' and 'quantifying in' are impossible across root clause boundaries.

And finally, we need an axiom for the **null discourse**, to ground the recursive construction of discourses:

$$\vdash \mathbf{e}\ ;\mathrm{D}\ ;\lambda_{c:\mathrm{c}}.c$$

Note that using the null discourse as the first premise in the continue rule yields the derived rule

$$\frac{\vdash s\ ;\mathrm{S}\ ;k}{\vdash s\ ;\mathrm{D}\ ;\mathsf{cc}\ k}\ ,$$

which says that any sentence can be 'promoted' to a single-sentence discourse.

5.5 The dynamic connectives

The content negation NOT A fundamental insight of dynamic semantics is that an *indefinite* inside the scope of negation *cannot* antecede definite anaphora in the *subsequent* discourse, but a *definite* in the scope of negation *can* be anteceded by an indefinite in *prior* discourse:

(3) a. Pedro has a donkey$_i$. It$_i$ $\left\{\begin{array}{l}\text{is}\\ \text{isn't}\end{array}\right\}$ friendly.

b. Pedro doesn't have a donkey$_i$. # It$_i$ $\left\{\begin{array}{l}\text{is}\\ \text{isn't}\end{array}\right\}$ friendly.

Such facts are accounted for by the following definition for the **content negation** NOT : $\mathrm{k} \rightarrow \mathrm{k}_0$:

$$\textsc{not} =_{\text{def}} \lambda_{k:\mathrm{k}}.\lambda_{c:\mathrm{c}}.\lambda_{\mathbf{x}^{|c|}}.\mathsf{not}\ \mathsf{exists}_{\mathbf{y}^{|k|}}.k\ c\ \mathbf{x},\mathbf{y}$$

This has the effect that DRs introduced within the scope of content negation are existentially bound within the scope of (static) propositional negation, and therefore inaccessible as antecedents for subsequent definite anaphora.

The effect of *double* content negation on any content k is to existentially bind any new DRs that it introduces:

$$\vdash \textsc{not}\ (\textsc{not}\ k) \equiv \lambda_{c:\mathrm{c}}.\lambda_{\mathbf{x}^{|c|}}.\mathsf{exists}_{\mathbf{y}^{|k|}}.k\ c\ \mathbf{x},\mathbf{y}$$

From this it follows that a content of degree 0 (i.e. one which introduces no new DRs) is equivalent to its own double negation. In particular, since the content negation of any content is itself of degree 0, content negation is equivalent to *triple* content negation.

The content conjunction AND Another fundamental insight of dynamic semantics is that sentential conjunction is not commutative:

(4) a. A farmer walked in and he sat down. (*Who is he?*)

b. He sat down and a farmer walked in. (*Who is he?*)

We capture this insight with the following definition of **content conjunction**, of type $\Pi_{h:\mathrm{k}}.\Pi_{k:\mathrm{k}}.\Pi_{c:\mathrm{c}}.\mathrm{c}_{|c|+|h|+|k|}$:

$$\textsc{and} =_{\text{def}} \lambda_{h:\mathrm{k}}.\lambda_{k:\mathrm{k}}.\lambda_{c:\mathrm{c}}.\lambda_{\mathbf{x}^{|c|},\mathbf{y}^{|h|},\mathbf{z}^{|k|}}.(h\ c\ \mathbf{x},\mathbf{y})\ \mathsf{and}\ (k\ (\mathsf{cc}\ h\ c)\ \mathbf{x},\mathbf{y},\mathbf{z})$$

Crucially, the input context ($\mathsf{cc}\ h\ c$) for the *second* conjunct is created by applying the update induced by the *first* conjunct to *its* input context c. It is not hard to show that the update induced by conjoined declarative utterances is the same as the function composition of the updates induced by the conjuncts:

$$\vdash \forall_{hk}.(\mathsf{cc}\ (h\ \textsc{and}\ k)) = \lambda_{c:\mathrm{c}}.\mathsf{cc}\ k\ (\mathsf{cc}\ h\ c)$$

The content disjunction OR As in other versions of dynamic semantics, we define **content disjunction** by DeMorgan duality:

$$\text{OR} =_{\text{def}} \lambda_{h:\text{k}}.\lambda_{k:\text{k}}.\text{NOT}\ ((\text{NOT}\ h)\ \text{AND}\ (\text{NOT}\ k))$$

This predicts that indefinites within either disjunct can't antecede subsequent definite anaphora:

(5) a. Either a donkey brayed or someone is making barnyard noises. # It's friendly.

b. Either someone is making barnyard noises, or a donkey brayed. # It's friendly.

c. # Either a donkey brayed, or it's friendly.

The content implication IMPLIES Our definition of **content implication** is modeled on a valid but nonstandard equivalence for static propositional implication:

$$\mathsf{implies} =_{\text{def}} \lambda_{pq}.(\mathsf{not}\ p)\ \mathsf{or}\ (p\ \mathsf{and}\ q)$$
$$\text{IMPLIES} =_{\text{def}} \lambda_{hk}.(\text{NOT}\ h)\ \text{OR}\ (h\ \text{AND}\ k)$$

This predicts that indefinites in either the antecedent or the consequent of the conditional cannot antecede definite anaphora in a subsequent sentence, but an indefinite in the antecedent *can* antecede definite anaphora in the consequent:

(6) a. If a donkey brayed, it's hungry. # We better feed it.

b. If Pedro is a farmer, he has a donkey. # He better feed it.

An additional virtue of this definition of content implication is that it gives rise to so-called *weak* readings of conditional sentences:

(7) a. If you have a donkey, I'll buy it.

b. If you have a donkey, I'll buy a donkey you have. (*weak*)

c. If you have a donkey, I'll buy every donkey you have. (*strong*).

There are known pragmatic strengthening strategies for inferring strong understandings from weak meanings, but if a strong semantics is used (say, based on a different static equivalence $\vdash (p\ \mathsf{implies}\ q) \equiv (\mathsf{not}\ (p\ \mathsf{and}\ (\mathsf{not}\ q)))$, then it is hard to explain where weak readings come from.

5.6 Dynamic generalized quantifiers (DGQs)

Three kinds of noun phrase We turn next to the dynamic meanings of indefinite referring expressions (e.g. *a donkey*), definite referring expressions (e.g. *it, the donkey*), and 'truly quantificational' NPs (e.g. *every donkey, no donkey*). Although these will all be of the same type q (DGQs), they differ radically in their discourse behavior. A use of an indefinite in an utterance introduces a new DR into its output context, while a definite 'picks up' or 'continues' an already existing DR:

(8) A donkey$_i$ brayed. It$_i$ was hungry.

But a use of a truly quantificational NP renders any DRs introduced within either its restriction or its scope inaccessible to the subsequent discourse:

(9) a. $\left\{\begin{matrix}\text{Every}\\ \text{No}\end{matrix}\right\}$ farmer that owned a donkey$_i$ was unhappy. # It$_i$ was lazy.

b. $\left\{\begin{matrix}\text{Every}\\ \text{No}\end{matrix}\right\}$ farmer owned a donkey$_i$. # It$_i$ was lazy.

However, an indefinite introduced in the restriction of a truly quantificational NP is accessible from its scope:

(10) $\left\{\begin{matrix}\text{Every}\\ \text{No}\end{matrix}\right\}$ farmer that owns a donkey$_i$ beats it$_i$.

Dynamic properties In static semantics, a GQ is a property of properties of entities (type $\mathrm{p}_1 \to \mathrm{p}$). Since, in dynamic semantics, n and u are the respective counterparts of static sense types e and p, it seems reasonable to assume that the type for the dynamic counterparts of properties of, and GQs over, entities would be, respectively, $\mathrm{n} \to \mathrm{k}$ and $(\mathrm{n} \to \mathrm{k}) \to \mathrm{k}$. However, some delicacy is called for, because we have to ensure that in applying a dynamic property to a DR, the resulting content is one which is defined on contexts which 'know about' the DR in question. Additionally, we have to take into consideration that a given dynamic property (e.g. *farmer that owns a donkey*) may itself introduce new DRs. Accordingly, we define (for each i) the type $\mathrm{d}_{1,i}$ of unary **dynamic properties** of degree i as

$$\mathrm{d}_{1,i} =_{\mathrm{def}} \mathit{\Pi}_{n:\mathrm{n}}.\mathit{\Pi}_{c:\mathrm{c}_{>n}}.\mathrm{c}_{|c|+i}\ ,$$

and the type of unary dynamic properties as the dependent sum

$$\mathrm{d}_1 =_{\mathrm{def}} \mathit{\Sigma}_{i:\mathrm{n}}.\mathit{\Pi}_{n:\mathrm{n}}.\mathit{\Pi}_{c:\mathrm{c}_{>n}}.\mathrm{c}_{|c|+i}\ .$$

In due course, we will define dynamic counterparts d_n for all the static types p_n.

Dynamic counterparts of static properties In dynamic semantics, the static-property senses of common nouns, predicative adjectives, and intransitive verbs have to be replaced by their dynamic counterparts, which are unary dynamic properties (of degree 0, since they introduce no DRs). This change is effected by the unary **dynamicization** function $\mathrm{dyn}_1 : \mathrm{p}_1 \to \mathrm{d}_{1,0}$ defined as follows:

$$\mathrm{dyn}_1 =_{\mathrm{def}} \lambda_{P:\mathrm{p}_1}.\lambda_{n:\mathrm{n}}.\lambda_{c:\mathrm{c}_{>n}}.\lambda_{\mathbf{x}^{|c|}}.P\ x_n$$

For example, the dynamic meaning of the common noun *donkey* is:

$$\textsc{donkey} =_{\mathrm{def}} (\mathrm{dyn}_1\ \mathsf{donkey}) = \lambda_{n:\mathrm{n}}.\lambda_{c:\mathrm{c}_{>n}}.\lambda_{\mathbf{x}^{|c|}}.\mathsf{donkey}\ x_n$$

This maps a DR n and a context c which knows about n to a content which asserts that (whichever entity corresponds to) n is a donkey. More generally, for each n : n, there is a type d_n of n-ary **dynamic relations**, the dynamic counterparts of the static types p_n. As in the unary case, each of these is a dependent sum

$$\mathrm{d}_n = \Sigma_{i:\mathrm{n}}.\mathrm{d}_{n,i}\ ,$$

where i ranges over the degree (number of newly introduced DRs) of the relation. For example, for $n = 2$:

$$\mathrm{d}_{2,i} =_{\mathrm{def}} \Pi_{m:\mathrm{n}}.\Pi_{n:\mathrm{n}}.\Pi_{c:\mathrm{c}_{>(\mathbf{max}\ m\ n)}}.\mathrm{c}_{|c|+i}$$

A special case of this is:

$$\mathrm{d}_{2,0} =_{\mathrm{def}} \Pi_{m:\mathrm{n}}.\Pi_{n:\mathrm{n}}.\Pi_{c:\mathrm{c}_{>(\mathbf{max}\ m\ n)}}.\mathrm{c}_{|c|}$$

Also as in the unary case, for each n : n, there is a function $\mathrm{dyn}_n : \mathrm{p}_n \to \mathrm{d}_n$ that maps each static relation to its dynamic counterpart. For $n < 3$, these are:

$$\begin{aligned}
\mathrm{dyn}_0 &=_{\mathrm{def}} \lambda_{p:\mathrm{p}}.\lambda_{c:\mathrm{c}}.\lambda_{\mathbf{x}^{|c|}}.p\\
\mathrm{dyn}_1 &=_{\mathrm{def}} \lambda_{P:\mathrm{p}_1}.\lambda_{n:\mathrm{n}}.\lambda_{c:\mathrm{c}_{>n}}.\lambda_{\mathbf{x}^{|c|}}.P\ x_n\\
\mathrm{dyn}_2 &=_{\mathrm{def}} \lambda_{R:\mathrm{p}_2}.\lambda_{m:\mathrm{n}}.\lambda_{n:\mathrm{n}}.\lambda_{c:\mathrm{c}_{>(\mathbf{max}\ m\ n)}}.\lambda_{\mathbf{x}^{|c|}}.R\ x_m\ x_n
\end{aligned}$$

For example:

$$\begin{aligned}
\textsc{rain} &= \lambda_{c:\mathrm{c}}.\lambda_{\mathbf{x}^{|c|}}.\mathsf{rain}\\
\textsc{donkey} &= \lambda_{n:\mathrm{n}}.\lambda_{c:\mathrm{c}_{>n}}.\lambda_{\mathbf{x}^{|c|}}.\mathsf{donkey}\ x_n\\
\textsc{own} &= \lambda_{m:\mathrm{n}}.\lambda_{n:\mathrm{n}}.\lambda_{c:\mathrm{c}_{>(\mathbf{max}\ m\ n)}}.\lambda_{\mathbf{x}^{|c|}}.\mathsf{own}\ x_m\ x_n
\end{aligned}$$

Indefinites To give a dynamic meaning for indefinites, we start with the **context extension** function $(\cdot)^+$ of type $\Pi_{c:\mathrm{c}}.\mathrm{c}_{|c|+1}$, defined as follows:

$$(\cdot)^+ =_{\mathrm{def}} \lambda_{c:\mathrm{c}}.\lambda_{\mathbf{x}^{|c|},y}.c\ \mathbf{x}$$

This just adds a new DR to any context. In terms of this, we now define the dynamic 'existential' quantifier EXISTS to be the following function of type $\Pi_{n:\mathrm{n}}.\Pi_{D:\mathrm{d}_{1,n}}.\mathrm{k}_{n+1}$ (i.e. it maps a dynamic property to a content of degree one greater than that of the dynamic property):

$$\textsc{exists} =_{\mathrm{def}} \lambda_{D:\mathrm{d}_1}.\lambda_{c:\mathrm{c}}.D\ \ |c|\ \ c^+$$

Then the dynamic meaning of the indefinite determiner is defined by analogy with its static counterpart:

$$\begin{aligned}
\mathsf{a} &=_{\mathrm{def}} \lambda_{PQ}.\mathsf{exists}_x.(P\ x)\ \mathsf{and}\ (Q\ x)\\
\textsc{a} &=_{\mathrm{def}} \lambda_{DE}.\textsc{exists}_n.(D\ n)\ \textsc{and}\ (E\ n)
\end{aligned}$$

For example, the (degree 1) content of *a donkey brays* is

$$\vdash \textsc{a donkey bray} = \lambda_{c:\mathrm{c}}.\lambda_{\mathbf{x}^{|c|},y}.(\mathsf{donkey}\ y)\ \mathsf{and}\ (\mathsf{bray}\ y)\ .$$

The fact that the variable y corresponding to the DR for the donkey is only λ-bound (not exists-bound) has as a consequence that this DR will be accessible to subsequent definite anaphora.

Definite anaphora To keep within space bounds, we provide here a simplified, Montague-like, treatment of anaphora in terms of lexical ambiguity as to which DR is the antecedent; [20] and [21] describe a mechanism (improving on the `sel` function of [11]) for selecting the (presupposed) DR that satisfies the definite expression's descriptive content. For example, the dynamic meaning of the n-th definite pronoun *it* is

$$\begin{aligned}\textsc{it}^n &=_{\mathrm{def}} \lambda_{D:\mathrm{d}_1}.\lambda_{c:\mathrm{c}_{>n}}.D\ n\ c\\ &= \lambda_{D:\mathrm{d}_1}.\lambda_{c:\mathrm{c}_{>n}}.\lambda_{\mathbf{x}^{|c|}}.D\ n\ c\ \mathbf{x}\ .\end{aligned}$$

Unlike an indefinite, which introduces a new DR, the definite simply resumes an old one.

A 'truly quantificational' DGQ Again by analogy with the static case, we define the dynamic universal quantifier FORALL as follows:

$$\begin{aligned}\mathsf{forall} &=_{\mathrm{def}} \lambda_P.\mathsf{not}\ (\mathsf{exists}_x.(\mathsf{not}\ (P\ x)))\\ \textsc{forall} &=_{\mathrm{def}} \lambda_D.\textsc{not}\ (\textsc{exists}_n.(\textsc{not}\ (D\ n)))\end{aligned}$$

Then, again analogizing to the static case, we define the dynamic universal determiner EVERY:

$$\begin{aligned}\mathsf{every} &=_{\mathrm{def}} \lambda_{PQ}.\mathsf{forall}_x.(P\ x)\ \mathsf{implies}\ (Q\ x)\\ \textsc{every} &=_{\mathrm{def}} \lambda_{DE}.\textsc{forall}_n.(D\ n)\ \textsc{implies}\ (E\ n)\end{aligned}$$

This in turn can be shown to be equivalent to

$$\lambda_{DE}.\textsc{not}\ (\textsc{exists}_n.((\textsc{not}\ (\textsc{not}(D\ n)))\ \textsc{and}\ (\textsc{not}\ ((D\ n)\ \textsc{and}\ (E\ n)))))$$

For example, a simple universal sentence like *Every donkey brays* ends up with the content EVERY DONKEY BRAY, which can be shown to be equivalent to

$$\lambda_{c:\mathrm{c}}.\lambda_{\mathbf{x}^{|c|}}.\mathsf{not}\ \mathsf{exists}_y.(\mathsf{donkey}\ y)\ \mathsf{and}\ (\mathsf{not}\ (\mathsf{bray}\ y))\ .$$

As desired, the fact that the variable y corresponding to the DR for the donkey is exists-bound (within the scope of negation) has as a consequence that this DR is *in*accessible to subsequent definite anaphora.

Donkey anaphora We conclude with a DyCG derivation for the classic universal donkey sentence *Every farmer that owns a donkey beats it.* Because we based our semantics for EVERY on the 'weak' dynamic implication IMPLIES, our analysis produces the weak reading (that every farmer that owns a donkey beats a donkey that s/he owns), which again is a desirable state of affairs (cf. [2, 16]). The lexical entries employed are:

$$\vdash \lambda_{sf}.f\ (\text{every} \cdot s)\ ;\ \text{N} \multimap (\text{NP} \multimap \text{S}) \multimap \text{S}\ ;\ \textsc{every}$$
$$\vdash \text{farmer}\ ;\ \text{N}\ ;\ \textsc{farmer}$$
$$\vdash \lambda_{sf}.s \cdot \text{that} \cdot (f\ \mathbf{e})\ ;\ \text{N} \multimap (\text{NP} \multimap \text{S}) \multimap \text{N}\ ;\ \textsc{that}$$
$$\vdash \lambda_{st}.s \cdot \text{owns} \cdot t\ ;\ \text{NP} \multimap \text{NP} \multimap \text{S}\ ;\ \textsc{own}$$
$$\vdash \lambda_{sf}.f\ (\text{a} \cdot s)\ ;\ \text{N} \multimap (\text{NP} \multimap \text{S}) \multimap \text{S}\ ;\ \textsc{a}$$
$$\vdash \text{donkey}\ ;\ \text{N}\ ;\ \textsc{donkey}$$
$$\vdash \lambda_{st}.s \cdot \text{beats} \cdot t\ ;\ \text{NP} \multimap \text{NP} \multimap \text{S}\ ;\ \textsc{beat}$$
$$\vdash \lambda_{f}.f\ \text{it}\ ;\ (\text{NP} \multimap \text{S}) \multimap \text{S}\ ;\ \textsc{it}^n$$

And the endsequent of the derivation is

$$\vdash \text{every} \cdot \text{farmer} \cdot \text{that} \cdot \text{owns} \cdot \text{a} \cdot \text{donkey} \cdot \text{beats} \cdot \text{it}\ ;\ \text{S}\ ;$$
$$\textsc{every}\ (\textsc{farmer that}\ \lambda_m.(\textsc{a donkey})_n.(\textsc{own}\ m\ n))\ \lambda_m.\textsc{it}^i.(\textsc{beat}\ m)\ .$$

In a given context, this will produce the (weak) donkey-anaphora reading provided i is selected to be whichever natural number corresponds to the DR introduced by *a donkey*. As mentioned above, [20] gives a more realistic treatment for the definite pronoun *it* as selecting the most salient, informationally unique discourse referent with the property of being nonhuman.

6 Conclusion

This dynamic semantics is not only fully compositional and expressed in pure (dependent) type theory, it also captures all of the central insights of the dynamic tradition, with indefinites introducing discourse referents, definites selecting their antecedents from the input context, and 'accessibility constraints' captured by existentially binding variables in the scope of negation and operators defined in terms of it. No idiosyncratic machinery is required, and we overcome the problem in which discourse contexts must have a certain arity by a mild use of dependent types, rather than extending the type theory with partial functions. As we have shown, this theory can be seen as a straightforward extension of the underlying static semantics, achieved by adding a type of discourse contexts and replacing entities with natural number indices into the list of discourse referents the incoming context 'knows about.' Since the underlying static semantics is hyperintensional, we avoid certain foundational problems with possible worlds approaches while allowing the grammar to focus on the dynamic senses of expressions, rather than bothering with worlds and extensions at them.

References

1. Beaver, D.I.: Presupposition and Assertion in Dynamic Semantics. CSLI Publications (2001)
2. Chierchia, G.: The Dynamics of Meaning: Anaphora, Presupposition, and the Theory of Grammar. University of Chicago Press (1995)
3. Church, A.: A formulation of the simple theory of types. Journal of Symbolic Logic 5(2), 56–68 (1940)
4. van Eijck, J., Unger, C.: Computational Semantics with Functional Programming. Cambridge University Press (2010)
5. Frege, G.: Über Sinn und Bedeutung. Zeitschrift für Philosophie und philosophische Kritik 100, 25–50 (1892), English translation titled *On Sense and Reference* in [7], pages 56–78
6. Gallin, D.: Intensional and Higher Order Modal Logic, Mathematics Studies, vol. 19. North-Holland, Amsterdam (1975)
7. Geach, P., Black, M.: Translations from the Philosophical Writings of Gottlob Frege. Blackwell, Oxford (1952)
8. Groenendijk, J., Stokhof, M.: Dynamic Montague grammar. In: Kálmán, L., Pólos, L. (eds.) Papers from the Second Symposium on Logic and Language. Akadémiai Kiadó (1990)
9. Groenendijk, J., Stokhof, M.: Dynamic predicate logic. Linguistics and Philosophy 14(1), 39–100 (1991)
10. de Groote, P.: Towards abstract categorial grammars. In: Association for Computational Linguistics, 39th Annual Meeting and 10th Conference of the European Chapter, Proceedings of the Conference (2001)
11. de Groote, P.: Towards a Montagovian account of dynamics. In: Proceedings of the 16th Conference on Semantics and Linguistic Theory (2006)
12. de Groote, P., Nederhof, M.J. (eds.): Formal Grammar. No. 7395 in Lecture Notes in Computer Science (2012)
13. Heim, I.: The Semantics of Definite and Indefinite Noun Phrases. Ph.D. thesis, University of Massachusetts, Amherst (1982)
14. Henkin, L.: Completeness in the theory of types. Journal of Symbolic Logic 15(2), 81–91 (1950)
15. Kamp, H.: A theory of truth and semantic representation. In: Groenendijk, J., Janssen, T., Stokhof, M. (eds.) Formal Methods in the Study of Language. Mathematical Center, Amsterdam (1981)
16. Kanazawa, M.: Weak vs. strong readings of donkey sentences and monotonicity inference in a dynamic setting. Linguistics and Philosophy 17(2), 109–158 (1994)
17. Kierstead, G., Martin, S.: A multistratal account of the projective Tagalog evidential 'daw'. In: Proceedings of the 22nd Conference on Semantics and Linguistic Theory (2012)
18. Kripke, S.: Naming and Necessity. Harvard University Press (1980)
19. Martin, S.: Weak familiarity and anaphoric accessibility in dynamic semantics. In: de Groote and Nederhof [12], pp. 287–306
20. Martin, S.: The Dynamics of Sense and Implicature. Ph.D. thesis, Ohio State University (2013)
21. Martin, S.: Supplemental update (In preparation), unpublished manuscript
22. Martin, S., Pollard, C.: A higher-order theory of presupposition. Studia Logica 100(4), 729–754 (2012)

23. Martin, S., Pollard, C.: Hyperintensional dynamic semantics: Analyzing definiteness with enriched contexts. In: de Groote and Nederhof [12], pp. 114–129
24. Montague, R.: The proper treatment of quantification in ordinary English. In: Thomason, R. (ed.) Formal Philosophy: Selected Papers of Richard Montague, pp. 247–270. Yale University Press, New Haven (1974)
25. Muskens, R.: Combining Montague semantics and discourse representation theory. Linguistics and Philosophy 19(2), 143–186 (1996)
26. Muskens, R.: Separating syntax and combinatorics in categorial grammar. Research on Language and Computation 5(3), 267–285 (2007)
27. Oehrle, R.T.: Term-labeled categorial type systems. Linguistics and Philosophy 17(6), 633–678 (1994)
28. Plummer, A., Pollard, C.: Agnostic possible worlds semantics. In: Béchet, D., Dikovsky, A. (eds.) Logical Aspects of Computational Linguistics. pp. 201–212. No. 7351 in Lecture Notes in Computer Science (2012)
29. Pollard, C.: Hyperintensions. Journal of Logic and Computation 18(2), 257–282 (2008)
30. Pollard, C.: Agnostic hyperintensional semantics. Synthese (in press)
31. Pollard, C., Yasavul, M.: Anaphoric clefts: the myth of exhaustivity (In preparation), to appear in *Proceedings of CLS 2014*
32. Thomason, R.: A model theory for propositional attitudes. Linguistics and Philosophy 4(1), 47–70 (1980)