EXTERNALLY-DYNAMIC DYNAMIC SEMANTICS[1]

PATRICK D. ELLIOTT

JULY 15, 2021

[1] This talk is based primarily on material from Elliott 2020b, where I develop a semantics based on the same principles as Externally-Dynamic Dynamic Semantics (EDS) for a first-order fragment. I'm grateful to Simon Charlow, Keny Chatain, Matthew Gotham, Julian Grove, Nathan Klinedinst, Matt Mandelkern, and Yasu Sudo.

1 Goals

There are two distinct but interrelated issues I'll address in this talk:

> Is the flow of referential information pre-compiled into logical operators (Heim 1982), or are the logical operators fundamentally truth-functional (Schlenker 2008, 2009)?

The perspective I'll develop here is that that referential information uniformly flows from left-to-right, as a byproduct of composition principles; the idiosyncratic control-flow associated with logical expressions is epiphenomenal.

> Just how successful are classical dynamic frameworks as a high-level description of possible anaphoric dependencies?

Even just in the domain of singular (in)definites, I'll argue that classical Dynamic Semantics (DS) ignores the influence of pragmatics, precluding a principled account of anaphora.

As a case study, we'll focus on *disjunction*, the treatment of which has proven to be especially problematic in classical DS.

(1) *Bathroom disjunctions*:
Either there is no^{1} bathroom, or it$_{1}$'s upstairs.

(2) *G&S disjunctions*:
Either a^{1} philosopher is giving a talk, or a^{1} linguist is.
(Either way) they$_{1}$'re very loud.

The alternative developed here, EDS, aside from being less stipulative than classical dynamic semantics, will have an improved empirical reach.

2 *Dynamic Semantics (DS)*

Content is influenced not just by *what* is said, but also by *how* it is said (Kamp 1981, Heim 1982).[2]

(3) a. Andrew has a child. She is at school.
b. Andrew is a parent. # She is at school.

In talking, we keep track not just of information concerning how things are, but also referential information concerning what/who the speaker intended to refer to; DS is therefore a theory of *aboutness*.

Part of the dynamic thesis is that uttering a sentence with an indefinite changes the referential information in the conversational context in a special way.

Sentences with pronouns are sensitive to referential information in a way which other expressions aren't.

Let's see how this is (classically) accomplished, by constructing a representative compositional dynamic fragment: DS.

2.1 *Discourse anaphora*

I'll start with a compositional take on the core ideas of Dynamic Predicate Logic (DPL) (Groenendijk & Stokhof 1991).[3] I'll call this DS.

DS will serve as a useful baseline comparison to EDS — the fragment outlined in §3.

In DS, sentences express relations on assignments.[4,5]

(4) $\mathbb{t} \coloneqq g \to \{\, g \,\}$

We can bootstrap a simple dynamic fragment by systematically identifying the propositional type with $\mathbb{t}$; predicates are functions from individuals to *tests*.[6]

(5) a. $[\![\text{swim}]\!] = \lambda x \,.\, \lambda g \,.\, \{\, g \mid \mathbf{swim}(x) \,\}$ $\quad e \to \mathbb{t}$
b. $[\![\text{hug}]\!] = \lambda x y \,.\, \lambda g \,.\, \{\, g \mid \mathbf{hug}(x)(y) \,\}$ $\quad e \to e \to \mathbb{t}$

Proper names receive their ordinary denotations, but both indefinites and pronouns take scope.

[2] The argument here relies on the fact that the initial sentences in (3a) and (3b) are contextually truth-conditionally equivalent. The requirement for an indefinite antecedent is sometimes called the *Formal Link Condition*.

[3] This is based primarily on the discussion in Charlow 2014, 2019b; see also Groenendijk & Stokhof 1990, Muskens 1996.

[4] We use right-associative arrow notation for function types (Carpenter 1998). $\{\,.\,\}$ is the constructor for *set types* The type of a dynamic proposition reflects the fact that meanings in DS are (i) environment-sensitive, and (ii) induce a non-deterministic update of the environment.

[5] For now, we'll simply assume that assignments are total functions whose domain is a finite set of variables.

[6] A dynamic proposition p is a *test*, iff for any assignment g, $p(g) = \{\, g \,\} \vee p(g) = \varnothing$.

(6) $[\![\text{Brian swims}]\!] = [\![\text{swim}]\!]\,([\![\text{Brian}]\!]) = \lambda g\,.\begin{cases} g & \textbf{swims}(\textbf{b}) \\ \varnothing & \text{otherwise} \end{cases}$

Indefinites induce *random assignment* wrt a variable n.[7]

(7) a. $[\![\text{some}^n\ \text{linguist}]\!] = \mathscr{E}^n(\textbf{ling})$
b. $\mathscr{E}^n(f) \coloneqq \lambda k\,.\bigcup_{f(x)} k(x)(g^{[n\mapsto x]})$ $\qquad \mathscr{E}^n : (e \to t) \to (e \to \Bbbt) \to \Bbbt$

E.g., let's say that Athulya and Brian are the linguists who swim:

(8) $[\![\text{some}^1\ \text{linguist swims}]\!] = \mathscr{E}^1(\textbf{ling})\,[\![\text{swims}]\!]$
$= \lambda g\,.\ \{\, g^{[1\mapsto \mathbf{a}]}, g^{[1\mapsto \mathbf{b}]} \,\}$

Pronouns read their value from the input assignment.

(9) $[\![\text{she}_n]\!] = \lambda k\,.\,\lambda g\,.\,k(g_n)(g)$ $\qquad (e \to \Bbbt) \to \Bbbt$

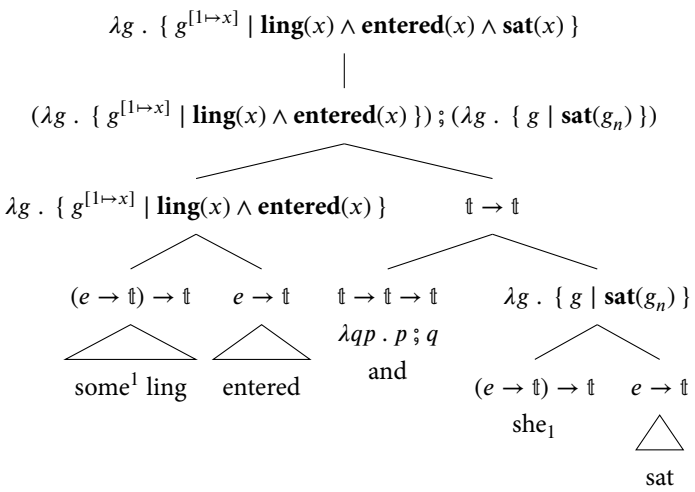
(10) $[\![\text{she}_1\ \text{swims}]\!] = [\![\text{she}_1]\!]\,([\![\text{swims}]\!]) = \lambda g\,.\ \{\, g \mid \textbf{swims}(g_1) \,\}$

In DS, conjunction denotes relational composition.

(11) a. $p\,;q \coloneqq \lambda g\,.\bigcup_{h\in p(g)} q(h)$ $\qquad (;) : \Bbbt \to \Bbbt \to \Bbbt$

We now have everything we need to account for discourse anaphora:

(12)



Captures the *internal* and *external dynamicism* of conjunction (Groenendijk & Stokhof 1991):

(13) a. Some^1 linguist entered and she$_1$ sat. She$_1$ fidgeted.
b. #She$_1$ entered and some^1 linguist sat.

[7] As a simplification, I assume that the restrictor of the indefinite simply denotes a set of individuals. This means that complex restrictors are out of reach for both DS and EDS. There is however not a principled reason why complex restrictors can't be incorporated, they merely lead to additional compositional complexity.

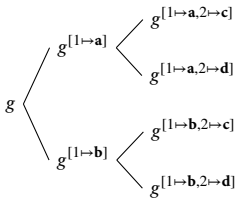


Figure 1: Composing relations on assignments

2.2 Closing off anaphoric possibilities

Negation

I'll start with an important auxiliary operator: dynamic closure tests whether or not a dynamic proposition has a non-empty output.[8]

(14) $\not\downarrow(p) \coloneqq \lambda g \,.\, p(g) \neq \varnothing$ $\qquad\qquad \mathbb{t} \rightarrow g \rightarrow t$

The standard dynamic entry for negation can be stated as a composition of closure and truth-functional negation **not** (Groenendijk & Stokhof 1990); "not ϕ" tests at g whether or not ϕ is true at g.[9]

(15) $[\![\text{not}]\!] = \lambda p \,.\, \lambda g \,.\, \{\, g \mid \textbf{not}(\not\downarrow(p)(g)) \,\} = \lambda g \,.\, \{\, g \mid p(g) = \varnothing \,\}$ $\qquad \mathbb{t} \rightarrow \mathbb{t}$

Result: a negative sentence is guaranteed to be a test.[10]

(16) $[\![\text{no}^1 \text{ linguist entered}]\!] = [\![\text{not}]\!]\,([\![\text{some}^1 \text{ linguist entered}]\!])$
$= \lambda g \,.\, \{\, g \mid \{\, x \mid \textbf{entered}(x) \,\} = \varnothing \,\}$

Prediction: an indefinite in the scope of negation is prevented from passing referential information to a subsequent pronoun ✓

(17) *No^1 linguist entered and she$_1$ sat.

Disjunction

The standard dynamic entry for disjunction (Groenendijk & Stokhof 1991) is also stated in terms of closure, alongside logical disjunction **or**.

(18) $[\![\text{or}_{\text{dyn}}]\!] = \lambda q \,.\, \lambda p \,.\, \{\, g \mid \not\downarrow(p)(g) \textbf{ or } \not\downarrow(p)(g) \,\}$ $\qquad\qquad \mathbb{t} \rightarrow \mathbb{t} \rightarrow \mathbb{t}$

Informally, at g "P or Q" tests if P is true or Q is true at g.

Dynamic disjunction is tailored to account for the (alleged) fact that disjunction in natural language apparently does not permit referential information to flow between disjuncts...

(19) #Either someone^1 is in the audience or they$_1$'re sitting down.

...or from out of disjunctions.[11]

[8] Dynamic closure gives us a notion of *truth at an assignment*.

[9] See also Rothschild 2017 and Gotham 2019 for discussion.

[10] Based on this property, Groenendijk & Stokhof (1991) classify negation as an *externally static* operator.

[11] Like negation, a disjunctive sentence in DS is guaranteed to be a test. On this basis, Groenendijk & Stokhof classify disjunction as *externally static*. Since referential information isn't passed between disjuncts, it is also *internally static*.

(20) Either this house hasn't been renovated, or there's a[1] bathroom.
It$_1$'s upstairs.

Importantly, DS also gives us the resources to state other kinds of disjunction, e.g., *program disjunction*.[12]

(22) $[\![\text{or}_{\text{prog}}]\!] = \lambda q\,.\,\lambda p\,.\,\{\, h \mid h \in p(g) \textbf{ or } h \in p(g)\,\}$ $\qquad\qquad \Uparrow \to \Uparrow \to \Uparrow$

Groenendijk & Stokhof (1991) propose program disjunction in order to account for sentences of the following kind:

(23) If Cathy cooks an[1] apple tart or an[1] apricot tart, it$_1$ will be delicious.

The fact that DS *allows* us to define both dynamic disjunction and program disjunction, and doesn't provide a principled way to decide between them, hints at a problem.

2.3 Problems

Contradictory empirical evidence pertaining to anaphora motivates an ad-hoc ambiguity between or_{dyn} and or_{prog}.

The dynamic entries for the logical operators can't be systematically deduced from their classical counterparts (Schlenker 2009, 2010).

More generally, DS arguably *strays too far from the classical*. For example, *double negation elimination* isn't valid (Groenendijk & Stokhof 1991), which seems like an undesirable result:

(24) John doesn't have no[1] shirt. He's wearing it$_1$.

In fact, it DS double negation is equivalent to closure ($\lightning$), so (24) is problematic[13]

The problem of Double-Negation Elimination (DNE) in another guise: Partee's *bathroom disjunctions*:

(25) Either there isn't any[1] bathroom, or it$_1$'s upstairs.

Relatedly, de Morgan's equivalences aren't valid in DS. Intuitively, (25) should be equivalent to the following:

[12] This is an especially clear demonstration of the fact that the truth-functional operators do not determine their DS correlates. Weirder kinds of disjunctions can also be easily defined, such as disjunctions that are only externally dynamic with respect to the right disjunct:

(21) $[\![\text{or}_{\text{R}}]\!] = \lambda q\,.\,\lambda p\,.\,\left\{ h \;\middle|\; \begin{array}{l}(h = g \wedge \lightning(p)(g)) \\ \textbf{or } h \in q(g)\end{array} \right\}$

[13] See Krahmer & Muskens 1995, Gotham 2019 for attempts to resolve this in a dynamic setting.

(26) It's not the case that there[1]'s a bathroom and it$_1$'s upstairs.

Neither dynamic disjunction nor program disjunction predict the possibility of any anaphoric dependency between disjuncts.

In the next section, we'll develop a system in which logical operators are just truth-functional operators — no dynamic entries will be necessary, with the important exception of indefinites.

Logical operators will be systematically lifted into the dynamic scaffolding via a constrained inventory of combinators.

3 *Externally-Dynamic Dynamic Semantics*

In the previous section, the only dynamic type was $\Bbbt$.

In EDS, we make use of a general recipe for constructing dynamic types (σ is an implicitly universally-quantified variable over types):[14]

(27) $D(\sigma) \coloneqq g \to \{ (\sigma, g) \}$

For example, sentences in EDS are type $D(t)$; VPs are typically of type $D(e \to t)$, etc.

The general strategy for constructing EDS is based on Charlow's (2019b) monadic grammar, although the result will be significantly different.

3.1 *Pronouns and partiality*

In EDS, pronouns are *dynamic individuals* of type $D(e)$. First, let's be more concrete about assignments.

In EDS, assignments are *partial*, i.e., potentially undefined for certain variables.

We'll model this by treating the domain of assignments (D_g) as a set of *total* functions $f : \mathbb{N} \to D_e$, where D_e contains a privileged individual # (the "impossible individual").[15]

Since EDS builds on a Strong Kleene logical foundation, we'll be making use of a *trivalent logic* with three truth values:

[14] Ultimately, everything will have to be intensionalized, in which case we keep track of worlds in both the input and output. For the time being, we abstract away from this dimension of meaning.

[15] For example, given a stock of variables $\{ n \mid 1 \leq n \leq 5 \}$, the following is a 'partial' assignment:

$$\begin{bmatrix} 1 \to \mathbf{a} \\ 2 \to \mathbf{b} \\ 3 \to \# \\ 4 \to \# \\ 5 \to \# \end{bmatrix}$$

(28) $D_t = \{ \textbf{yes}, \textbf{no}, \textbf{maybe} \}$

We'll introduce an operator δ to model the presuppositions of pronouns, with the following semantics:

(29) $$\delta(x) = \begin{cases} \textbf{maybe} & x = \# \\ \textbf{yes} & \text{else} \end{cases}$$

By stipulation, predicates return **maybe** if any of their arguments are #.[16,17]

(30) a. $[\![\text{swim}]\!] = \lambda x \,.\, \delta(x) \,\&\, \textbf{swim}(x)$ $\quad e \to t$

b. $[\![\text{hug}]\!] = \lambda xy \,.\, \delta(y) \,\&\, \delta(x) \,\&\, \textbf{hug}(x)(y)$ $\quad e \to e \to t$

Pronouns have the following semantics; they read their value off the input:

(31) $[\![\text{she}_n]\!] = \lambda g \,.\, \{ (g_n, g) \}$ $\quad D(e)$

Sentences with a pronoun indexed n denote dynamic propositions (type $D(t)$) which, given an input assignment g, pair output assignments with **maybe** if $g_n = \#$:

(32) $[\![\text{she}_n \text{ swims}]\!] = \lambda g \,.\, \{ (\delta(g_n) \,\&\, \textbf{swims}(g_n), g) \}$ $\quad D(t)$

Alternatively:

(33) $$\lambda g \,.\, \{ (\textbf{yes}, g) \mid g_n \neq \# \wedge \textbf{swims}(g_n)) \} \cup \{ (\textbf{no}, g) \mid g_n) \neq \# \wedge \neg(\textbf{swims}(g_n)) \} \cup \{ (\textbf{maybe}, g) \mid g_n = \# \}$$

Consider what happens if we interpret "she swims" relative to the unique initial assignment:[18]

(34) $[\![\text{she}_1 \text{ swims}]\!] (g_\top) = \{ (\delta(\#) \,\&\, \textbf{swims}(\#), g_\top) \} = \{ (\textbf{maybe}, g_\top) \}$

N.b. unlike in DS, where falsity corresponds to an empty output, here we keep track of truth + referential information.

3.2 *Indefinites*

The semantics of indefinite NPs will crucially be a departure from the classical.

[16] We use & in the meta-language for Weak Kleene conjunction.

[17] We'll often suppress conjuncts whose job is to check definedness of arguments in what follows, but bear in mind that if any of a predicates arguments are #, the result is **maybe**.

[18] $g_\top$ maps every variable to the impossible individual. The initial assignment has a privileged role to play in the pragmatics of anaphora, since we can model an *initial context* in which nothing has been uttered as $C \times \{ g_\top \}$. This will be relevant later on.

Random assignment in EDS

In order to understand how indefinites work in EDS, it will be helpful to first define random assignment relative to a restrictor r, as a straightforward adaptation of DS random assignment.

(35) $\varepsilon^n = \lambda r\,.\,\lambda k\,.\,\lambda g\,.\,\bigcup_{r(x)} k(x)(g^{[n\mapsto x]}) \qquad (e \to D(t)) \to D(t)$

(36) $\varepsilon^n(\textbf{ling})(\lambda x\,.\,\lambda g\,.\,\{\,(\textbf{swim}(x), g)\,\}) = \lambda g\,.\,\bigcup_{\textbf{ling}(x)} \{\,(\textbf{swim}(x), g^{[n\mapsto x]})\,\}$

Equivalently:

(37) $\lambda g\,.\,\{\,(\textbf{yes}, g^{[n\mapsto x]}) \mid \textbf{ling}(x) \wedge \textbf{swim}(x)\,\}$
$\cup\,\{\,(\textbf{no}, g^{[n\mapsto x]}) \mid \textbf{ling}(x) \wedge \neg(\textbf{swim}(x))\,\}$

We take an input assignment g, extend g indeterministically at n to linguists, and:

- Tag those assignments extended to a linguist who swims with **yes**.
- Tag those assignments extended to a linguist who doesn't swim with **no**.

We'll also define a notion which will come in handy in a number of different places: the *polarized referential information* of a sentence, which we write as $\mathscr{R}$. Given an input assignment, this is simply the set of output assignments associated with a particular truth-value.

(38) $\mathscr{R}^g_t(p) = \{\,h \mid (t, h) \in p(g)\,\} \qquad t \to D(t) \to \mathbb{t}$

We can immediately use this notion to define truth at a point:

(39) a. $\textbf{true}(p)(g) \coloneqq \mathscr{R}^g_+(p) \neq \varnothing \qquad D(t) \to g \to t$
b. $\textbf{false}(p)(g) \coloneqq \mathscr{R}^g_+(p) = \varnothing \wedge \mathscr{R}^g_-(p) \neq \varnothing$
c. $\textbf{neither}(p)(g) \coloneqq \mathscr{R}^g_+(p) = \varnothing \wedge \mathscr{R}^g_-(p) = \varnothing \wedge \mathscr{R}^g_u(p) \neq \varnothing$

Our semantics for indefinites will crucially be stated in terms of a *positive closure* operator †, which ensures that, if its prejacent isn't true with respect to the input assignment, referential information is filtered out:

(40) $\dagger(p)(g) \coloneqq \{\,(\textbf{yes}, h) \mid h \in \mathscr{R}^g_+(p)\,\} \qquad D(t) \to D(t)$
$\cup\,\{\,(\textbf{no}, g) \mid \textbf{false}(p)(g)\,\}$
$\cup\,\{\,(\textbf{maybe}, g) \mid \textbf{neither}(p)(g)\,\}$

N.b., we have the following logical truth, for all dynamic propositions p.[19]

[19] This is obvious, since positive closure explicitly leaves positive referential information unchanged.

(41) $\mathscr{R}^g_+(\dagger(p)) \equiv \mathscr{R}^g_+\, p$

We'll define our semantics for indefinites as the composition of random assignment and positive closure:

(42) $[\![\text{some}^n \text{ linguist}]\!] = \lambda k\ .\ \dagger(\varepsilon^n(\mathbf{ling})(k)) \qquad (e \to D(t)) \to D(t)$

Let's consider what this means for the semantics of sentences with indefinites:

(43) $$\begin{aligned}[\![\text{some}^n \text{ linguist swims}]\!] &= \dagger(\varepsilon^n(\mathbf{ling})(\lambda x\ .\ \lambda g\ .\ \{\ (\mathbf{swim}(x), g)\ \})) \\ &= \lambda g\ .\ \{\ (\mathbf{yes}, g^{[n\to x]})\mid \mathbf{ling}(x) \wedge \mathbf{swim}(x)\ \} \\ &\qquad \cup \{\ (\mathbf{no}, g)\mid \neg\exists x[\mathbf{ling}(x) \wedge \mathbf{swim}(x)]\ \}\end{aligned}$$

Assignments are indetermistically extended at n to linguists who swim and paired with **yes**; if there aren't any linguists who swim, the input assignment is simply paired with **no**.[20]

[20] It's probably already clear that EDS admits of an alternative *multivalent* presentation, in which each sentence is paired with a positive, negative, and undefined extension, relative to an input (see, e.g, Krahmer & Muskens 1995, van den Berg 1996 for this kind of setup). This however obscures the fact that EDS is constructed by dynamicizing, in a quite general fashion, a trivalent logical substrate.

3.3 *Compositionality*

As a reminder, EDS is a fragment in which predicates, proper names, logical expressions etc. have their classical semantics. This is a methodological assumption.

(44) a. $[\![\text{John}]\!] = \mathbf{John}$ $\qquad e$
b. $[\![\text{swim}]\!] = \lambda x\ .\ \delta(x)\ \&\ \mathbf{swim}(x)$ $\qquad e \to t$
c. $[\![\text{not}]\!] = \mathbf{not}$ $\qquad t \to t$

The intuition here is that only a sub-part of the grammar wears its dynamic capabilities on its sleeve.

In order to lift expressions without inherent dynamics into EDS, we're going to need three combinators, which together with D constitute an *applicative functor*.[21]

[21] AKA an *idiom*, AKA a *lax monoidal functor*. See appendix B for details.

η (pronounced *pure*) lifts an a into a (trivially) dynamic a.

(45) $\eta(a) \coloneqq \lambda g\ .\ \{\ (a, g)\ \} \qquad \sigma \to D(\sigma)$

Forward dynamic function application (//) performs Function Application (FA) while threading referential information from the left-to-right; this is essentially a generalization of dynamic conjunction.

(46) $m_{D(\sigma\to\tau)} /\!/ n_{D(\sigma)} \coloneqq \lambda g\ .\ \bigcup_{(f,h)\in m(g)} \{\ (f(x), i)\mid (x, i) \in n(h)\ \}$

Backward dynamic function application (⧵⧵) performs backward FA while threading referential information from left-to-right.

(47) $m_{D(\sigma)} \mathbin{\backslash\!\backslash} n_{D(\sigma\to\tau)} \coloneqq \lambda g \,.\, \bigcup_{(x,h)\in m(g)} \{\, (f(x), i) \mid (x, i) \in n(h) \,\}$

Both ⫽ and ⧵⧵ are necessary to capture the intuition that the dynamics of referential information is sensitive to linear order, rather than function-argument relations.

We can define a useful combinator in terms of η and ⫽, which we'll use to map functions into dynamic arguments — *map* (◊):

(48) $f_{\sigma\to\tau} \Diamond m_{D(\sigma)} \coloneqq \eta(f) \mathbin{/\!\!/} m$
$\Rightarrow \lambda g \,.\, \{\, (f(x), h) \mid (x, h) \in m(g) \,\}$

First, we need to make sure that the flow of referential information mirrors linear order. There are a number of different ways of doing this.[22]

We'll cleave to an interpretive architecture (Heim & Kratzer 1998) and assume the following additional composition principles, alongside ◊ and ordinary FA:[23]

(49) $\left[\!\!\left[\begin{array}{ccc} & \gamma & \\ \alpha_{D(\sigma\to\tau)} & & \beta_{D(\sigma)} \end{array} \right]\!\!\right] \coloneqq [\![\alpha]\!] \mathbin{/\!\!/} [\![\beta]\!]$

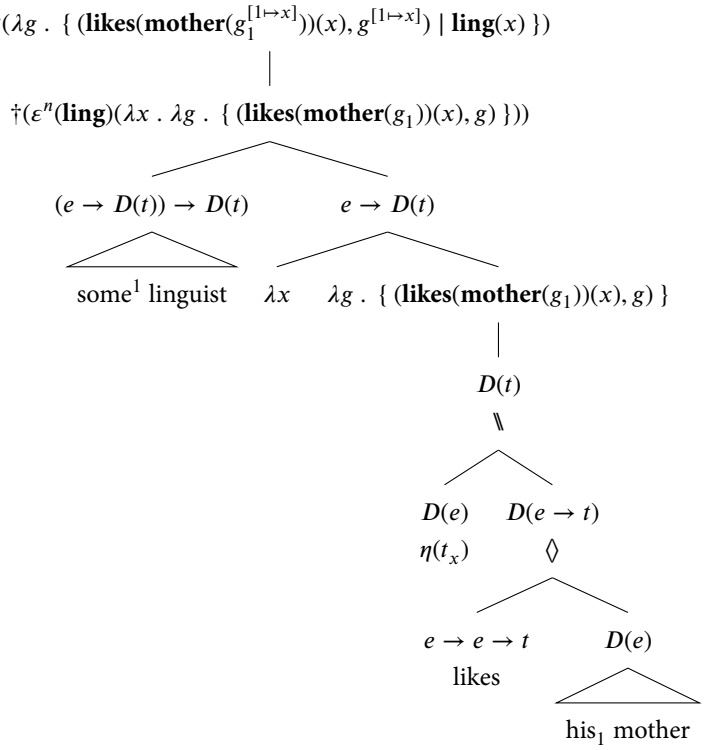
(50) $\left[\!\!\left[\begin{array}{ccc} & \gamma & \\ \alpha_{D(\sigma)} & & \beta_{D(\sigma\to\tau)} \end{array} \right]\!\!\right] \coloneqq [\![\alpha]\!] \mathbin{\backslash\!\backslash} [\![\beta]\!]$

Although we'll be exclusively concentrating on the sentential level, an immediate consequence of this regime is the possibility of in-scope dynamic binding.

[22] One elegant possibility is using the directed categories of categorial grammar (Carpenter 1998).

[23] Note that an obvious consequence of generalizing DS to a compositional setting is that semantic interpretation *must* be sensitive to linear order. In an interpretive setting, the respective order of α and β should be pre-encoded in the syntax.

(51)



3.4 Lifting logical operators

The following schema will fall out on the basis of the demands of the compositional system (desugaring ◊):

(52)

a.	$\lambda p \,.\, \eta(\textbf{not}) \mathbin{/\!\!/} p$	$D(t) \to D(t)$
b.	$\lambda q \,.\, \lambda p \,.\, p \mathbin{\backslash\!\!\backslash} (\eta(\textbf{and}) \mathbin{/\!\!/} q)$	$D(t) \to D(t) \to D(t)$
c.	$\lambda q \,.\, \lambda p \,.\, p \mathbin{\backslash\!\!\backslash} (\eta(\textbf{or}) \mathbin{/\!\!/} q)$	$D(t) \to D(t) \to D(t)$

Recall that we're assuming three truth values **yes**, **no**, and **maybe**. For the logical operators, we'll assume a *strong Kleene* semantics.

Let's start with the simplest case negation (we'll write **not** for strong Kleene negation):

not	
yes	**no**
no	**yes**
maybe	**maybe**

Figure 2: Negation in strong Kleene

(53) $\textbf{not} : t \to t$

Just as with other functions, sentential negation ($t \to t$) composes with a dynamic proposition ($D(t)$) via ◊.

When we apply negation to a sentence with an indefinite, truth-values in the output set are simply flipped.

(54) $[\![\text{no}^1 \text{ linguist swims}]\!] = \textbf{not} \Diamond [\![\text{some}^1 \text{ linguist swims}]\!]$
$= \lambda g \,.\, \{ (\textbf{not}(\textbf{yes}), g^{[n \mapsto x]}) \mid \textbf{ling}(x) \wedge \textbf{swim}(x) \}$
$\cup \{ (\textbf{not}(\textbf{no}), g) \mid \neg \exists x[\textbf{ling}(x) \wedge \textbf{swim}(x)] \}$
$= \lambda g \,.\, \{ (\textbf{no}, g^{[n \mapsto x]}) \mid \textbf{ling}(x) \wedge \textbf{swim}(x) \}$
$\cup \{ (\textbf{yes}, g) \mid \neg \exists x[\textbf{ling}(x) \wedge \textbf{swim}(x)] \}$

What this means is that, if the negative sentence is *true*, no referential information is introduced. This captures the *external staticity* of negation.

(55) John doesn't have a^1 shirt. # It$_1$'s in the closet.

Moreover, we can now see why random assignment isn't fit for purpose as a semantics of indefinites in EDS — this is because, if we simply lift logical negation into EDS it commutes with simple random assignment $\epsilon^n(f)$ (verification is left as an exercise).

Indefinites in EDS however don't commute with negation:

(56) $[\![\text{a}^1 \text{ linguist doesn't swim}]\!] = [\![\text{a}^1 \text{ linguist}]\!] (\lambda x \,.\, \eta([\![\text{not } t_x \text{ swims}]\!]))$
$= \lambda g \,.\, \{ (\textbf{yes}, g^{[1 \mapsto x]}) \mid \textbf{ling}(x) \wedge \neg(\textbf{swim}(x)) \}$
$\cup \{ (\textbf{no}, g) \mid \exists x[\textbf{ling}(x) \wedge \textbf{swim}(x)] \}$

Since all negation does is flip the polarity of referential information in the output, it's obvious that the following is a logical-truth about its behavior in this system:

(57) For any dynamic proposition p
not $\Diamond$ (**not** $\Diamond$ p) $\equiv p$

Double-negation elimination is therefore valid, and the problem of double-negation is thereby addressed (cf. Gotham 2019).[24]

[24] Modulo a putative uniqueness effect. See Appendix A.

(58) a. John doesn't have NO^1 shirt. It$_1$'s in his closet.
b. John has a^1 shirt. It$_1$'s in his closet.

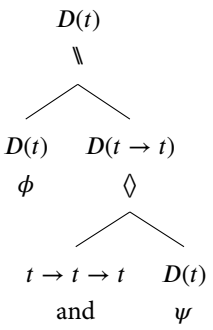
3.5 *Conjunction and discourse anaphora*

Our composition principles also allow binary truth-functional operators to compose with dynamic propositions, such that the flow of referential information tracks the linear order of the juncts.

$\wedge_s$	yes	no	maybe
yes	yes	no	maybe
no	no	no	no
maybe	maybe	no	maybe

Figure 3: Conjunction in strong Kleene

(59)



Crucially *Egli's theorem*[25] holds in this system, at least with respect to the *positive* referential information conveyed by the sentence.[26]

[25] In FoL terms, Egli's theorem states that $\exists^n(\phi) \wedge \psi \equiv \exists^n(\phi \wedge \psi)$

[26] More generally $\mathscr{R}^g_+([\![\phi \text{ and } \psi]\!]) = (\lambda g\ .\ \mathscr{R}^g_+([\![\phi]\!])) \fatsemi (\lambda g\ .\ \mathscr{R}^g_+([\![\psi]\!]))$.

(60) $[\![\text{a}^1 \text{ linguist entered and she}_1 \text{ sat}]\!]$

$= [\![\text{a}^1 \text{ linguist entered}]\!] \setminus\!\setminus (\Diamond([\![\text{and}]\!])([\![\text{she}_1 \text{ sat}]\!]))$

$= \lambda g\ .\ \{\ (t \wedge_s u, i) \mid \exists h[(t, h) \in [\![\text{a}^1 \text{ ling entered}]\!]\ (g) \wedge (u, i) \in [\![\text{she}_1 \text{ sat}]\!]\ (h)]\ \}$

$= \lambda g\ .\ \{\ (\mathbf{yes}, g^{[1 \mapsto x]}) \mid \mathbf{ling}(x) \wedge \mathbf{entered}(x) \wedge \mathbf{sat}(x)\ \}$

$\cup\ \{\ (\mathbf{no}, g^{[1 \mapsto x]}) \mid \mathbf{ling}(x) \wedge \mathbf{entered}(x) \wedge \neg(\mathbf{sat}(x))\ \}$

$\cup\ \{\ (\mathbf{no}, g) \mid \neg\exists x[\mathbf{ling}(x) \wedge \mathbf{entered}(x)]\ \}$

Another way of thinking about it:

Scenario 1: there is a linguist who entered. The first conjunct introduces a *positive* discourse referent — the second disjunct retains the positive discourse referent if the linguist sat, and makes it negative otherwise. We never have to consider any **maybe** values.

(61) $\lambda g\ .\ \{\ (\mathbf{yes} \wedge_s u, h) \mid \exists x[\mathbf{ling}(x) \wedge \mathbf{entered}(x) \wedge (u, h) \in \{\ (\mathbf{sat}(x), g^{[1 \mapsto x]})\ \}]\ \}$

Scenario 2: there is no linguist who entered. The second conjunct never effects the truth-value (thanks to Strong Kleene conjunction), nor introduces any discourse referents. **Maybe** values don't affect the falsity of the conjunctive sentence.

(62) $\lambda g\ .\ \{\ (\mathbf{no} \wedge_s u, h) \mid (u, h) \in \{\ (\mathbf{sat}(g_1), g)\ \}\ \}$

3.6 Bathroom disjunctions

Just like conjunctive sentences, disjunctive sentences compose via $\Diamond$ and $\setminus\!\setminus$.

Let's begin by considering what's predicted for a *bathroom disjunction*.

$\vee_s$	**yes**	**no**	**maybe**
yes	**yes**	**yes**	**yes**
no	**yes**	**no**	**maybe**
maybe	**yes**	**maybe**	**maybe**

Figure 4: Disjunction in strong Kleene

(63) $[\![$either there's no^1 bathroom or it$_1$'s upstairs$]\!]$

$[\![$there's no^1 bathroom$]\!] \Cap (\Diamond([\![$or$]\!])([\![$it$_1$'s upstairs$]\!]))$

$\lambda g\ .\ \{\ (t \vee_s u, i) \mid \exists h[(t,h) \in [\![$there's no^1 bathroom$]\!]\ (g) \wedge (u,i) \in [\![it_1$'s upstairs$]\!]\ (h)]\ \}$

$\lambda g\ .\ \{\ (\textbf{yes}, g) \mid \neg\exists x[\textbf{bathroom}(x)]\ \}$

$\cup\ \{\ (\textbf{yes}, g^{[1\mapsto x]}) \mid \textbf{bathroom}(x) \wedge \textbf{upstairs}(x)\ \}$

$\cup\ \{\ (\textbf{no}, g^{[1\mapsto x]}) \mid \textbf{bathroom}(x) \wedge \neg(\textbf{upstairs}(x))\ \}$

Another way of thinking about it:

Scenario 1: there's no bathroom. The second disjunct never effects the truth-value (thanks to Strong Kleene disjunction), nor introduces any discourse referents.

(64) $\lambda g\ .\ \{\ (\textbf{yes} \vee_s u, h) \mid (u,h) \in \{\ (\textbf{upstairs}(g_1), g)\ \}\ \}$

Scenario 2: There is a bathroom. The first disjunct introduces a *negative* discourse referent — the second disjunct makes the discourse referent positive if the bathroom is upstairs, and negative otherwise.

(65) $\lambda g\ .\ \{\ (\textbf{no} \vee_s u, h) \mid \exists x[\textbf{bathroom}(x) \wedge (u,h) \in \{\ (\textbf{upstairs}(x), g^{[1\mapsto x]})\ \}]\ \}$

The problem of bathroom disjunctions is resolved.

3.7 Donkey sentences

Although certainly false, let's entertain as an idealization the possibility that natural language conditionals can be modeled using material implication **if**…**then** ($t \to t \to t$).

$\to_s$	yes	no	maybe
yes	yes	no	maybe
no	yes	yes	yes
maybe	yes	maybe	maybe

Figure 5: Disjunction in strong Kleene

We didn't talk about donkey sentences in the previous section, but one of the centerpieces of classical DS is a treatment of dynamic implication that ascribes strong *universal* truth-conditions to donkey sentences.

(66) If anyone^1 is outside, then they$_1$ are happy.

Skipping over the details, we make the following predictions for donkey sentences:

(67) $[\![\text{If anyone}^1 \text{ is outside, then they}_1 \text{ are happy}]\!]$
$= [\![\text{if anyone}^1 \text{ is outside}]\!] \setminus\!\!\setminus (\Diamond([\![\text{if...then}]\!])\,[\![\text{they}_1\text{'re happy}]\!])$
$= \lambda g\,.\ \{\,(t \rightarrow_s u, i) \mid \exists h[(t,h) \in [\![\text{anyone}^1 \text{ is outside}]\!]\,(g) \wedge (u,i) \in [\![\text{they}_1\text{'re happy}]\!]\,(h)]\,\}$
$= \lambda g\,.\ \{\,(\textbf{yes}, g^{[1\mapsto x]}) \mid \textbf{outside}(x) \wedge \textbf{happy}(x)\,\}$
$\cup\ \{\,(\textbf{no}, g) \mid \neg\exists x[\textbf{outside}(x)]\,\}$
$\cup\ \{\,(\textbf{no}, g^{[1\mapsto x]}) \mid \textbf{outside}(x) \wedge \neg(\textbf{happy}(x))\,\}$

Another way of thinking about:

Scenario 1: someone is outside. The antecedent introduces a *positive* discourse referent — the consequent makes the discourse referent positive if they are happy, and negative if not.

(68) $\lambda g\,.\ \{\,(\textbf{yes} \rightarrow_s u, h) \mid \exists x[\textbf{outside}(x) \wedge (u,h) \in \{\,(\textbf{happy}(x), g^{[1\mapsto x]})\,\}]\,\}$

Scenario 2: nobody is outside. The consequent never effects the truth-value, nor introduces any discourse referents:

(69) $\lambda g\,.\ \{\,(\textbf{no} \rightarrow_s u, h) \mid (u,h) \in \{\,(\textbf{happy}(g_1), g)\,\}\,\}$

Prediction: donkey sentences have weak, existential truth-conditions, i.e., (66) is true just so long as someone is outside and happy; the existence of someone outside who is unhappy doesn't falsify the sentence, under this reading.

This is indeed attested, the following can be true even if Gennaro has multiple credit cards, just so long as he paid with one of them (Chierchia 1995).

(70) If Gennaro has a credit card, he paid with it.

WiP: how to derive the strong reading. At worst, EDS is on a par with classical dynamic theories in this respect, which can only derive the strong reading.[27]

[27] Elliott (2020b) explores the possibility of deriving the strong reading as an implicature, via innocent exclusion (Bar-Lev & Fox 2017, Bar-Lev 2018). Any theory of donkey sentences that derives the weak reading as the basic reading must be supplemented with a means of deriving the strong reading. See, e.g., Champollion, Bumford & Henderson 2019 for another recent approach to the strong/weak distinction.

4 *Pragmatics, and the problem of too many discourse referents*

The moniker EDS was chosen because nothing in the semantics of the logical operators prevents referential information from being passed forward.

This means that, e.g., disjunctive sentences are both externally and internally *dynamic* as far as the semantics is concerned.

To see the problem, consider the following:

(71) Either this house hasn't been renovated, or there's a^1 bathroom.
It$_1$'s upstairs.

Suppose there is in fact exactly one bathroom *b*. The disjunctive sentence will introduce a positive *bathroom* discourse referent, and we predict anaphora to be licensed, contrary to fact.

A similar problem arises with material implication and negated conjunctions (left as an exercise).

As we've seen however, we don't want to build external staticity into the semantics of disjunction, as this leads to a dilemma, prompting the introduction of program disjunction.

In order to chart a way out, we'll build on an observation by Rothschild (2017).

4.1 Contextual entailment and anaphora

In a discourse with an asserted disjunctive sentence, if the truth of the disjunct containing an indefinite is later contextually entailed, anaphora becomes possible (Rothschild 2017).

Context: *the director of a play (A) has lost track of time, and doesn't know what day it is. The director is certain, however, that on Saturday and Sunday, different critics will be in the audience, and utters the disjunctive sentence in (72a). A's assistant (B), knows what day it is, and utters the sentence in (73b), which contextually entails the second disjunct. Subsequently, anaphora is licensed in (72c).*

(72) a. A: Either it's a weekday, or a^1 critic is watching our play.
b. B: It's Saturday.
c. A: They$_1$'d better give us a good review.

We can make the same point for conditionals.

(73) a. A: If it's the weekend, then a[1] critic is watching our play.
b. B: It's Saturday.
c. A: They$_1$'d better give us a good review.

Resolution: complex sentences can give the illusion of external staticity, given the conversational backgrounds against which they can be felicitously uttered.

4.2 Anaphora in a Stalnakerian pragmatics

In order to make good on this intuition, we'll formalize a Stalnakerian pragmatics to supplement EDS.

First, we systematically intensionalize the fragment, by adding a world parameter: a dynamic σ is a function from a world-assignment pair, to a σ-world-assignment triple.

(74) $D(\sigma) \coloneqq (s, g) \to \{ (\sigma, s, g) \}$

World-sensitive expressions (i.e., predicates) are lifted into D via the following modified η−operator:

(75) $\pi(a) \coloneqq \lambda(w, g)\ .\ \{ (a(w), w, g) \}$ $\qquad\qquad$ $\pi : (s \to \sigma) \to D(\sigma)$

Everything else remains the same, modulo some minor tweaks to keep track of world parameters.

We'll assume a relatively standard dynamic notion of an information state, consisting of a set of world-assignment pairs.

Definition 4.1 (Information state)**.** An *information state c* is a set of world-assignment pairs. Where:

- $c_\top$, the initial information state, is defined as: $W \times \{ g_\top \}$.
- $c_\varnothing$, the absurd information state is the empty set $\varnothing$

Now we define an *update* operation to model the effect on a context (which we model as an information state) of asserting a sentence. As usual, update is assumed to be subject to Stalnaker's *bridge principle* (von Fintel 2008), generalized to information states in the obvious way.

Definition 4.2 (Update)**.**

$$c[\phi] := \begin{cases} \bigcup\limits_{(w,g)\in c} \mathscr{R}_{+}^{w,g}([\![\phi]\!]) & \forall (w,g) \in c[\textbf{true}([\![\phi]\!])(w,g) \vee \textbf{false}([\![\phi]\!])(w,g)] \\ \varnothing & \text{otherwise} \end{cases}$$

Immediate consequence: Heim's familiarity presupposition is derived.

4.3 External staticity via ignorance

Disjunctive sentences place a requirement on the context — an utterance of a sentence of the form "P or Q" is only felicitous if both P and Q are *real* possibilities, i.e., the context shouldn't entail the truth/falsity of either of the disjuncts.

(76) Context: *it's common ground that someone was in the audience.*
Either someone was in the audience or the event was a disaster.

We can use this fact to account for the apparent external staticity of disjunction. Consider the following space of logical possibilities, representing a conversational background against which the disjunctive sentence may be uttered:

- w_{ad}: a was in the audience, and the event was a disaster.
- $w_{a\neg d}$: a was in the audience, and the event wasn't a disaster.
- $w_{\varnothing d}$: nobody was in the audience, and the event was a disaster.
- $w_{\varnothing \neg d}$: nobody was in the audience, and the event wasn't a disaster.

And consider the sentence under consideration:

(77) Either someone^{1} was in the audience, or the event was a disaster.

The positive referential information associated with the disjunctive sentence:

(78) $\{ (w, g^{[1\mapsto x]}) \mid \textbf{audience}_{w}(x) \}$
$\cup \{ (w,g) \mid \neg\exists x[\textbf{audience}_{w}(x)] \wedge \textbf{disaster}_{w}(\textbf{event}) \}$

We can now consider the result of updating the initial information state with the disjunctive sentence. Note that the bridge principle is trivially satisfied, since the sentence doesn't contain any free variables.

(79) $\left\{ \begin{array}{l} (w_{ad}, [1 \mapsto a]), \\ (w_{a\neg d}, [1 \mapsto a], \\ (w_{\varnothing d}, g_\top), \end{array} \right\}$

The resulting information state is one in which 1 is *not familiar*! This means that the presupposition of a subsequent sentence with a matching free variable won't be satisfied.

This account correctly captures the contextual entailment facts: an intermediate assertion can eliminate the world-assignment pair $(w_\varnothing, g_\top)$, thus rendering 1 familiar.[28]

(80) *Context: total ignorance*
   a. Either someone[1] was in the audience, or the event was a disaster.
   b. (Actually) the event wasn't a disaster.
   c. So, I hope she$_1$ enjoyed it.

[28] I'm optimistic that this general style of explanation can be extended to the (apparent) external staticity of conditional sentences. but this is complicated by the fact that material implication is undoubtedly not a realistic semantic proposal for conditional sentences of English.

The examples motivating program disjunction already follow straightforwardly from EDS. To illustrate, consider the following:

(81) Either a[1] linguist is here, or a[1] philosopher is.

The union of the two different ways of dynamically verifying the disjunctive sentence (81) gives us its positive extension. The salient point to note here is that the output set *only* contains assignments at which 1 is defined.

To my knowledge, this constitutes the first analysis of disjunction in dynamic semantics which straightforwardly captures both bathroom disjunctions and examples motivating program disjunction in a straightforward fashion.

4.4 Extensions

See Elliott 2020b for a similar account of other restrictions on anaphora:

- Restrictions on anaphora from negated conjunctions are accounted for also via ignorance inferences: utterances of the form "not (P and Q)" imply that "not P" and "not Q" are real possibilities.
- The *internal staticity* of disjunction is derived via a dynamic formulation of

Hurford's constraint.

5 Conclusion

We've achieved a dynamic semantics which is up-front about what exactly it stipulates.

Concretely, the locus of stipulation is in the compositional rules, which we stipulate pass referential information from left-to-right.

The idea is that there is a single switch which gives rise to incrementality in anaphoric processing; this isn't localized to the lexical entries of individual connectives.[29]

[29] This makes clear, and unexplored predictions with respect to acquisition, and cross-linguistic uniformity with respect to referential information flow.

In developing a more principled theory of anaphora, what we've learned is that the literature has essentially been mistaken in taking the accessibility generalizations at face value.

In order to maintain a parsimonious semantic theory, due care needs to be taken to address the role of pragmatic factors.

Developing an understanding of the pragmatics of referential information is essential in order to improve on our understanding of the semantic component.

As we've seen, it's possible to retain some of the appealing aspects of dynamic semantics - such as the dynamic notion of content - while improving upon the stipulative nature of extant dynamic theories.

References

Asudeh, Ash & Gianluca Giorgolo. 2020. *Enriched Meanings: Natural Language Semantics with Category Theory* (Oxford Studies in Semantics and Pragmatics). Oxford, New York: Oxford University Press. 208 pp.

Bar-Lev, Moshe E. 2018. *Free choice, homogeneity, and innocent inclusion*. The Hebrew University of Jerusalem dissertation.

Bar-Lev, Moshe E. & Danny Fox. 2017. Universal free choice and innocent inclusion. In *Proceedings of SALT 27*.

Carpenter, Bob. 1998. *Type-logical semantics* (Language, Speech, and Communication). Cambridge, Mass: MIT Press. 575 pp.

Champollion, Lucas, Dylan Bumford & Robert Henderson. 2019. Donkeys under discussion. *Semantics and Pragmatics* 12(0). 1. https://semprag.org/index.php/sp/article/view/sp.12.1 (11 November, 2020).

Charlow, Simon. 2014. *On the semantics of exceptional scope*. New Brunswick: Rutgers University dissertation.

Charlow, Simon. 2019a. A modular theory of pronouns and binding. Unpublished manuscript. Rutgers University. https://ling.auf.net/lingbuzz/003720.

Charlow, Simon. 2019b. Static and dynamic exceptional scope. lingbuzz/004650.

Chierchia, Gennaro. 1995. *Dynamics of meaning - anaphora, presupposition, and the theory of grammar*. Chicago: University of Chicago Press. 270 pp.

Elliott, Patrick D. 2020a. A flexible scope theory of intensionality. lingbuzz/005107. MIT. https://ling.auf.net/lingbuzz/005107.

Elliott, Patrick D. 2020b. Towards a principled logic of anaphora. lingbuzz/005562. MIT. https://ling.auf.net/lingbuzz/005562. Submitted to Semantics & Pragmatics.

Gotham, Matthew. 2019. Double negation, excluded middle and accessibility in dynamic semanttics. In Julian J. Schlöder, Dean McHugh & Floris Roelofsen (eds.), *Proceedings of the 22nd Amsterdam Colloquium*, 142–151.

Groenendijk, Jeroen & Martin Stokhof. 1990. *Dynamic Montague Grammar*. Report. https://eprints.illc.uva.nl/id/eprint/1148/ (2 July, 2021).

Groenendijk, Jeroen & Martin Stokhof. 1991. Dynamic predicate logic. *Linguistics and Philosophy* 14(1). 39–100.

Heim, Irene. 1982. *The semantics of definite and indefinite noun phrases*. University of Massachusetts - Amherst dissertation.

Heim, Irene. 1991. Artikel und definitheit. In Armin von Stechow & Dieter Wunderlich (eds.), *Semantik: Ein internationales Handbuch der zeitgenössischen Forschung*, 487–535. de Gruyter Mouton.

Heim, Irene & Angelika Kratzer. 1998. *Semantics in generative grammar* (Blackwell Textbooks in Linguistics 13). Malden, MA: Blackwell. 324 pp.

Kamp, Hans. 1981. A theory of truth and semantic representation. In Paul Portner & Barbara H. Partee (eds.), *Formal semantics: The essential readings*, 189–222. Blackwell.

Kobele, Gregory M. 2018. The Cooper Storage Idiom. *Journal of Logic, Language and Information* 27(2). 95–131. https://doi.org/10.1007/s10849-017-9263-1 (21 April, 2021).

Krahmer, Emiel & Reinhard Muskens. 1995. Negation and Disjunction in Discourse Representation Theory. *Journal of Semantics* 12(4). 357–376. https://academic.oup.com/jos/article/12/4/357/1728199 (27 June, 2020).

Mcbride, Conor & Ross Paterson. 2008. Applicative programming with effects. *Journal of Functional Programming* 18(1). http://www.journals.cambridge.org/abstract_S0956796807006326.

Muskens, Reinhard. 1996. Combining Montague semantics and discourse representation. *Linguistics and Philosophy* 19(2). 143–186. https://doi.org/10.1007/BF00635836 (8 July, 2021).

Rothschild, Daniel. 2017. A trivalent approach to anaphora and presupposition. In Alexandre Cremers, Thom van Gessel & Floris Roelofsen (eds.), *Proceedings of the 21st Amsterdam Colloquium*, 1–13.

Schlenker, Philippe. 2008. Be Articulate: A pragmatic theory of presupposition projection. *Theoretical Linguistics* 34(3). https://www.degruyter.com/view/j/thli.2008.34.issue-3/thli.2008.013/thli.2008.013.xml (13 September, 2020).

Schlenker, Philippe. 2009. Local contexts. *Semantics and Pragmatics* 2. http://semprag.org/article/view/sp.2.3.

Schlenker, Philippe. 2010. Local contexts and local meanings. *Philosophical Studies: An International Journal for Philosophy in the Analytic Tradition* 151(1). 115–142.

Sudo, Yasu. 2020. Scalar implicatures with discourse referents: A case study on the plurality inference of plural nouns. Unpublished manuscript. UCL. https://ling.auf.net/lingbuzz/005602.

van den Berg, M. H. 1996. Some aspects of the internal structure of discourse. The dynamics of nominal anaphora. https://dare.uva.nl/search?arno.record.id=7073 (31 August, 2020).

von Fintel, Kai. 2008. What Is Presupposition Accommodation, Again?*. *Philosophical Perspectives* 22(1). 137–170. https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1520-8583.2008.00144.x (18 November, 2020).

Wadler, Philip. 1995. Monads for functional programming. In *Advanced functional programming* (Lecture Notes in Computer Science), 24–52. Springer, Berlin, Heidelberg. https://link.springer.com/chapter/10.1007/3-540-59451-5_2.

A *Uniqueness and maximality*

Gotham (2019: p144) claims that both doubly-negated indefinites and bathroom disjunctions give rise to uniqueness inferences, unlike their positive counterparts:

(82) a. It's not true that John doesn't own a[1] shirt. It$_1$'s in the wardrobe.
b. Either John doesn't own a[1] shirt, or it$_1$'s in the wardrobe.
⇝ *If John owns a shirt, he owns exactly one*

(83) John has a shirt. It$_1$'s in the wardrobe.
↛ *If John owns a shirt, he owns exactly one*

This contrast is of course a problem for the *classicality* of EDS.

One obvious response is to challenge the judgement in (83) — it seems to me that even in a positive context, we observe a uniqueness inference (cf. Heim 1982).[30]

This dovetails with Sudo's observation that singular indefinites sometimes give rise to maximality (i.e., uniqueness) effects (Sudo 2020), much like plural pronouns.

(85) This coat has a[1] pocket. It_1 is inside. ⇝ *this coat has exactly one pocket*

One straightfoward way of capturing maximality inferences with singular indefinites is to modify the semantics of pronouns.

B Applicative functors

A tuple $(f, \texttt{pure}, \star)$ is an *applicative functor* (Mcbride & Paterson 2008) iff:

- f is a function from *types to types*.
- `pure` turns expressions of type σ into expressions of type $f(\sigma)$ (for every type σ).
- $\star$ allows expressions of type $f(\sigma \to \tau)$ to be treated as functions from $f(\sigma)$ to $f(\tau)$ (for all types σ, τ).
- `pure` and $\star$ satisfy the *applicative functor laws* (detailed below).

The intuition behind applicative functors is that we have some object of type σ somehow embedded in an object of a richer type $f(\sigma)$. Many operations are defined over objects of the simpler type σ; we'd like to systematically derive the *insensitivity* of (much of) our grammar to the rich structure provided by f. An applicative functor does exactly this.[31]

To qualify as an applicative functor `pure` and `apply` must satisfy the *applicative laws* given below:

(86)
a. $\lambda m \,.\, \texttt{pure}(id) \star m = id$ **Identity**
b. $\texttt{pure}(\circ) \star u \star v \star w = u \star (v \star w)$ **Composition**
c. $\texttt{pure}(f) \star \texttt{pure}(x) = \texttt{pure}(f(x))$ **Homomorphism**
d. $u \star \texttt{pure}(x) = \texttt{pure}(\lambda f \,.\, f(x)) \star u$ **Interchange**

Here I informally prove the applicative laws for $(D, \eta, /\!/)$, the applicative functor underlying EDS.

[30] It's not clear how else we might explain why the following sentence strongly implies that John has only one ear:

(84) #John has an ear.

[31] Applicative functors can be factored out of much extant compositional machinery — for concrete examples, see e.g., Charlow (2019a) on assignment-sensitivity, Kobele (2018) on Cooper storage, and Elliott (2020a) on world-sensitivity.

A related yet more powerful notion is that of a *monad* (Wadler 1995), which furnishes an applicative functor with an additional operation `join`, which collapses expressions of type $f(f(\sigma))$ to expressions of type $f(\sigma)$. Monads have been explicitly applied to linguistic phenomena by, e.g., Charlow (2014), Asudeh & Giorgolo (2020). Unlike monads, applicative functors *compose*, automatically providing theories with a degree of modularity.

Identity: we start with the left-hand of the formula, and demonstrate that it's equivalent to id.

(87) a. $\lambda m\,.\,\eta(id) \mathbin{/\!\!/} m$
b. $\Rightarrow \lambda m\,.\,(\lambda g\,.\,\{\,(id, g)\,\}) \mathbin{/\!\!/} m$
c. $\Rightarrow \lambda m\,.\,(\lambda g\,.\,\{\,(id, g)\,\}) \mathbin{/\!\!/} m$
d. $\Rightarrow \lambda m\,.\,\lambda g\,.\,\{\,(f(x), i) \mid \exists h[(f, h) \in \{\,(id, g)\,\} \wedge (x, i) \in m(h)]\,\}$
e. $\Rightarrow \lambda m\,.\,\lambda g\,.\,\{\,(id(x), i) \mid (x, i) \in m(g)\,\}$
f. $\Rightarrow \lambda m\,.\,\lambda g\,.\,\{\,(x, i) \mid (x, i) \in m(g)\,\}$
g. $\Rightarrow \lambda m\,.\,m$

Composition: we start with the left-hand side of the formula:

(88) a. $\eta(\circ) \mathbin{/\!\!/} u \mathbin{/\!\!/} v \mathbin{/\!\!/} w$
b. $\eta(\lambda f\,.\,\lambda f'\,.\,\lambda x\,.\,f(f'(x))) \mathbin{/\!\!/} u \mathbin{/\!\!/} v \mathbin{/\!\!/} w$
c. $(\lambda g\,.\,\{\,(\lambda f\,.\,\lambda f'\,.\,\lambda x\,.\,f(f'(x)), g)\,\}) \mathbin{/\!\!/} u \mathbin{/\!\!/} v \mathbin{/\!\!/} w$
d. $(\lambda g\,.\,\{\,(\lambda f'\,.\,\lambda x\,.\,f(f'(x)), h) \mid (f, h) \in u(g)\,\}) \mathbin{/\!\!/} v \mathbin{/\!\!/} w$
e. $(\lambda g\,.\,\{\,(\lambda x\,.\,f(f'(x)), i) \mid \exists h[(f, h) \in u(g) \wedge (f', i) \in v(h)]\,\}) \mathbin{/\!\!/} w$
f. $\lambda g\,.\,\{\,(f(f'(x)), j) \mid \exists h, i[(f, h) \in u(g) \wedge (f', i) \in v(h) \wedge (x, j) \in w(i)]\,\}$

Now we move to the right hand side of the formula and show that it reduces to the same result:

(89) a. $u \mathbin{/\!\!/} (v \mathbin{/\!\!/} w)$
b. $u \mathbin{/\!\!/} (\lambda h\,.\,\{\,(f'(x), j) \mid \exists h[(f', i) \in v(h) \wedge (x, j) \in w(i)]\,\})$
c. $\lambda g\,.\,\{\,(f(f'(x)), j) \mid \exists h, i[(f, h) \in u(g) \wedge (f', i) \in v(h) \wedge x(j) \in w(i)]\,\}$

Homomorphism: we start with the left-hand side of the formula, and demonstrate that it's equivalent to the $\eta(f(x))$.

(90) a. $\eta(f) \mathbin{/\!\!/} \eta(x)$
b. $(\lambda g\,.\,\{\,(f, g)\,\}) \mathbin{/\!\!/} \eta(x)$
c. $(\lambda g\,.\,\{\,(f, g)\,\}) \mathbin{/\!\!/} (\lambda g'\,.\,\{\,(x, g')\,\})$
d. $\lambda g\,.\,\{\,(p(a), i) \mid \exists h[(p, h) \in \{\,(f, g)\,\} \wedge (a, i) \in \{\,(x, h)\,\}]\,\}$
e. $\lambda g\,.\,\{\,(f(x), g)\,\}$

Interchange: we start with the left-hand side of the formula:

(91) a. $u \mathbin{/\!\!/} \eta(x)$
b. $u \mathbin{/\!\!/} (\lambda g\,.\,\{\,(x, g)\,\})$
c. $\lambda g\,.\,\{\,(f(x), h) \mid (f, h) \in u(g)\,\}$

Now we move to the right-hand side:

(92) a. $\eta(\lambda f \,.\, f(x)) \mathbin{/\!\!/} u$
b. $\lambda g \,.\, \{\, (\lambda f \,.\, f(x), g) \,\} \mathbin{/\!\!/} u$
c. $\lambda g \,.\, \{\, ((\lambda f \,.\, f(x))(f), h) \mid (f, h) \in u(g) \,\}$
d. $\lambda g \,.\, \{\, (f(x), h) \mid (f, h) \in u(g) \,\}$