Externally-dynamic dynamic semantics

Patrick D. Elliott

February 21, 2022

Contents

1 Externally-dynamic dynamic semantics

In the first generation dynamic systems we've considered, culminating in pointwise FCS, dynamics are a sentential phenomenon.

Charlow teaches us how to factor out dynamics into our compositional regime. We'll make use of this in EDS (Charlow 2014, 2020).

First, we adopt Charlow's general recipe for a dynamic type.[1, 2]

(1) $D\ a := g \to \{\, (a \times g) \,\}$

For example, sentences in EDS will be type $D\ t$; VPs in EDS will be type $D\ (e \to t)$.

[1] a is an implicitly universally-quantified variable over types.
[2] Initially, we'll present EDS as an extensional system; ultimately, everything will need to be intensionalized.

1.1 Pronouns and partiality

In EDS, much like in Charlow's monadic grammar, pronouns are expressions of type $D\ e$, i.e., *dynamic individuals*.

In EDS, assignments are assumed to be *partial*, i.e., undefined for certain variables.

We'll model this by treating the domain of assignments (D_g) as a set of *total* functions $f : V \to D_e$, where D_e contains a privileged value $\#_e$ - the impossible individual.[3]

For example, given a stock of variables $\{ x, y, z \}$, the following is a partial assignment:

(2) $$\begin{bmatrix} x & \to \textbf{josie} \\ y & \to \textbf{sarah} \\ z & \to \#_e \end{bmatrix}$$

The unique initial assignment, $g_\top$, maps every $v \in V$ to the impossible individual.

Pronouns have the following semantics in EDS:

(3) $\textbf{she}_v := \lambda g\,.\,\{ (g_v, g) \}$ $\qquad D\ e$

Since EDS builds on a Strong Kleene logical foundation, we'll make use of three distinct truth values:

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

We'll make use of an operator $\delta : t \to t$ to model presuppositions, with the following semantics.

(5) $$\delta(t) = \begin{cases} \textbf{yes} & t = \textbf{yes} \\ \textbf{maybe} & \text{otherwise} \end{cases}$$

Sentences with a pronoun indexed v presuppose that v is defined at the input assignment. Formally:

(6) $\textbf{she}_v\ \textbf{satDown} := \lambda g\,.\,\{ (\delta(g_v \neq \#_e)\ \&\ \textbf{satDown}(g_v), g) \}$

An alternative rendering:

(7) $$\begin{aligned} &\lambda g\,.\,\{ (\textbf{yes}, g) \mid \textbf{satDown}(g_v) \wedge g_v \neq \#_e \} \\ &\quad \cup \{ (\textbf{no}, g) \mid \neg\textbf{satDown}(g_v) \wedge g_v \neq \#_e \} \\ &\quad \cup \{ (\textbf{maybe}, g) \mid g_v = \#_e \} \end{aligned}$$

[3] See (Mandelkern 2022) for a similar set up.

1.2 Indefinites

1.2.1 Random assignment in EDS

It will be helpful to first define the correlate of DPL *random assignment* in EDS (relative to a restrictor r).

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

Let's see this in action (importantly, this is **not** our entry for the indefinite determiner).

(9) $\varepsilon^v(\mathbf{ling})(\lambda x\,.\,\lambda g\,.\,\{\,(\mathbf{swims}(x), g)\,\})$ $\qquad D\ t$

(10) $\lambda g\,.\,\{\,(\mathbf{swim}(x), g^{[v\to x]}) \mid \mathbf{ling}(x)\,\}$

An equivalent, illuminating rendering:

(11) $\lambda g\,.\,\{\,(\mathbf{yes}, g^{[v\to x]}) \mid \mathbf{ling}(x) \wedge \mathbf{swim}(x)\,\} \cup \{\,(\mathbf{no}, g^{[v\to x]}) \mid \mathbf{ling}(x) \wedge \mathbf{swim}(x)\,\}$

We take an input assingment g, and extend g indeterministically at v (DPL style) 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 define an auxiliary notion now which will come in handy in a few different places: the *polarized anaphoric information* of a sentence relative to an assignment, which we'll write as $\mathbf{A}^+_g / \mathbf{A}^-_g$.

(12) $\mathbf{A}^+_g(p) := \{\,h \mid (\mathbf{yes}, h) \in p(g)\,\}$

(13) $\mathbf{A}^-_g(p) := \{\,h \mid (\mathbf{no}, h) \in p(g)\,\}$

We can use this notion to provide an intuitive definition of truth at a point: a sentence is *true* wrt an assignment g if there is some way of verifying p at g, *false* if there is no way of verifying p at g, but some way of falsifying p at g, and neither true nor false otherwise.

(14) $\mathbf{true}_g(p) := \mathbf{A}^+_g(p) \neq \emptyset$

(15) $\mathbf{false}_g(p) := \mathbf{A}^+_g(p) = \emptyset \wedge \mathbf{A}^-_g(p) \neq \emptyset$

(16) $\mathbf{neither}_g(p) := \mathbf{A}^+_g(p) = \emptyset \wedge \mathbf{A}^-_g(p) = \emptyset$

Finally, we state our *positive closure operator* $\dagger_g$, which will be crucially implicated in our semantics for the indefinite article.

The positive closure operator only allows anaphoric information to pass through if its argument is classically true.

(17) $\dagger(p)(g) := \{\, (\mathbf{yes}, h) \in p(g) \,\} \cup \{\, (\mathbf{no}, g) \mid \mathbf{false}_g(p) \,\} \cup \{\, (\mathbf{maybe}, g) \mid \mathbf{neither}_g(p) \,\}$

The following is a logical truth in EDS (dagger elimination):

(18) $\mathbf{A}^+_g(\dagger(p)) = \mathbf{A}^+_g(p)$

Now we can state our final proposal for the semantics of indefinites as the composition of random assignment and positive closure.

(19) $\mathbf{a.ling}^v := \lambda k \,.\, \dagger(\varepsilon^v(\mathbf{ling})(k))$ $\qquad (e \to D\ t) \to D\ t$

(20) $\mathbf{a.ling}^v\ (\lambda x \,.\, \lambda g \,.\, \{\, (\mathbf{swim}(x), g) \,\})$

(21) $= \dagger(\varepsilon^v(\mathbf{ling})(\lambda x \,.\, \lambda g \,.\, \{\, (\mathbf{swim}(x), g) \,\}))$

(22) $= \lambda g \,.\, \{\, (\mathbf{yes}, g^{[v \to x]}) \mid \mathbf{ling}(x) \wedge \mathbf{swim}(x) \,\} \cup \{\, (\mathbf{no}, g) \mid \neg \exists x [\mathbf{ling}(x) \wedge \mathbf{swim}(x)] \,\}$

The input assignment is indeterministically extended at v to linguists who swim, and paired with **yes**; if there aren't any linguists who swim, the input assignment is paired with **no**.

1.3 Compositionality

As a methodological principle, we'll insist that proper names, predicates, logical expressions etc. don't have any inherent dynamics.

(23) $\mathbf{John} : t$

(24) $\mathbf{swim} : e \to t$

(25) $\mathbf{not} : t \to t$

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 need just three combinators, which constitute an *applicative functor*.

Pure (η) lifts any expression a into a trivially dynamic a.

(26) $\eta(a) := \lambda g\,.\,\{\,(a, g)\,\} \qquad\qquad \eta : a \to D\ a$

Dynamic FA (//) does function application and threads anaphoric information from left-to-right.

(27) $m\ //\ n := \lambda g\,.\,\bigcup_{(f,g')\in m(g)} \{\,(f(x), g'') \mid (x, g'') \in n(g')\,\} \qquad (//) : D\ (a \to b) \to D\ a \to D\ b$

Dynamic backwards FA (\\\) does backwards function application and threads anaphoric information from left-to-right.

(28) $m \setminus\setminus n := \lambda g\,.\,\bigcup_{(x,g')\in m(g)} \{\,(f(x), g'') \mid (f, g'') \in n(g')\,\} \qquad (\setminus\setminus) : D\ a \to D\ (a \to b) \to D\ b$

Composition:

(29) $\left[\!\left[\begin{array}{ccc} & \gamma & \\ \alpha_{D\ (a\to b)} & & \beta_{D\ a} \end{array} \right]\!\right] = [\![\alpha]\!]\ //\ [\![\beta]\!]$

(30) $\left[\!\left[\begin{array}{ccc} & \gamma & \\ \alpha_{D\ a} & & \beta_{D\ (a\to b)} \end{array} \right]\!\right] = [\![\alpha]\!] \setminus\setminus [\![\beta]\!]$

Note that i'm assuming that the flow of anaphoric information is conditioned by linear order, but a different assumption is just a matter of adjusting the rules stated above (cf. (Privoznov 2021)).

Some exercises - note that in-scope dynamic binding follows immediately from the composition principles and our semantics for indefinites (which extends DPL-style random assignment).

(31) $\mathbf{she}_v \setminus\setminus \eta(\mathbf{sat.down}) = \lambda g\,.\,\{\,(\mathbf{satDown}(g_v), g)\,\}$

(32) $\mathbf{a.ling}^v\ (\lambda x\,.\,\eta(\mathbf{walked.in}(x))) = \dagger(\varepsilon^v(\mathbf{ling})(\lambda x\,.\,\lambda g\,.\,\{\,(\mathbf{walked.in}(x), g)\,\}))$

(33) $\mathbf{a.ling}^v\ (\lambda x\,.\,\eta(\mathbf{introduced(j)})//(\mathbf{her_v\,.\,mother})) = \dagger(\varepsilon^v(\mathbf{ling})(\lambda x\,.\,\lambda g\,.\,\{\,(\mathbf{introduce}(j)(\mathbf{mother.of}\ g_v)(x$

1.4 Lifting logical operators

1.4.1 Strong Kleene

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

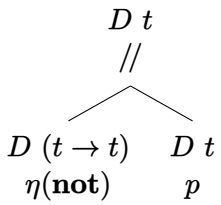
$\wedge_s$	**yes**	**no**	**maybe**
yes	**yes**	**no**	**maybe**
no	**no**	**no**	**no**
maybe	**maybe**	**no**	**maybe**

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

$\rightarrow_s$	**yes**	**no**	**maybe**
yes	**yes**	**no**	**maybe**
no	**yes**	**yes**	**yes**
maybe	**yes**	**maybe**	**maybe**

Figure 1: Strong Kleene truth tables

1.4.2 Negation



2 References

References

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

Charlow, Simon. 2020. Static and dynamic exceptional scope. Unpublished manuscript. Accepted at journal of semantics.

Mandelkern, Matthew. 2022. Witnesses. *Linguistics and Philosophy*.

Privoznov, Dmitry. 2021. Spelling Spell Out out. A friendly syntactic amendment to dynamic semantics. Unpublished manuscript. MIT. `https://ling.auf.net/lingbuzz/006279`.