Conventional Implicature as a scope phenomenon

Simon Charlow

Rutgers, The State University of New Jersey

Workshop on Continuations & Scope
May 22, 2015 · NYU

Goals for today

- Give a semantics for non-restrictive relatives, where supplemental content interacts with its semantic context by **taking scope**.
- Concretely, I'll extend the dynamic semantics of Charlow 2014 with some apparatus for two-dimensional content, à la Potts 2005.
- I argue that the extension is in fact **all we need to say** – the empirical properties we're after just fall out.

Where we are

Data

Composition: dynamic side effects

Adding in CIs

Accounting for our data

Wrapping up

Basic data

- Today we'll be looking at non-restrictive relative clauses (**NRRC**s):

(1) Sue, who's smart, bribed Jon, who isn't.

- Two related questions:
  - What sort of meaning should we assign this sentence?
  - How is the NRRC compositionally integrated?

Non-interaction

- In general, content introduced by NRRCs seems not to interact with other operators in a sentence:

(2) I didn't read *Beowulf*, which is a stone-cold classic.

(3) If John, who likes dancing, comes, the party will be great.

- Reminiscent of presupposition, but ultimately distinct (e.g., as Potts 2005 points out, the NRRC's content can't be presupposed!).

Scope of the anchor

- Scope of the anchor (after AnderBois et al. 2015: ex.72):

  (4) John didn't read a book, which Mary had recommended.

- This sentence **only** allows a wide-scope reading for the indefinite.

Quantifiers not welcome

- Quantifiers cannot serve as anchors:

(5) I read {*Beowulf*, a book, *no book}, which Mary likes.

Binding

- An indefinite can bind out of a NRRC (e.g. AnderBois et al. 2015):

(6) John, who nearly killed a woman$_i$ with his car, visited her$_i$ in the hospital.

- In fact, binding can go both ways:

(7) A boy$_i$ read Beowulf, which he$_i$ loved.

- Quantifiers, alas, still not allowed:

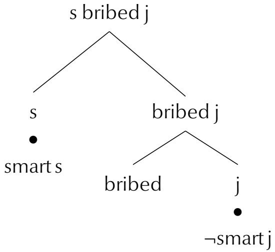
(8) *John, who nearly killed no woman$_i$ with his car, visited her$_i$ in the hospital.

(9) *No boy$_i$ read Beowulf, which he$_i$ loved.

- Surprising: we see the NRRC semantically interacting with the rest of the sentence, despite apparent non-interaction observed prior.

Potts 2005: radical separation

- Potts-style **parsetree** for *Sue, who's smart, bribed Jon, who isn't*:



- This representation is interpreted by pruning the bulleted meanings and conjoining them in a separate dimension.
- Worries: non-compositional (à la e.g. DRT), how to get interaction between the two dimensions? What is special about indefinites?

AnderBois et al. 2015: no separation

- No distinguished dimension for supplemental content.
- Distinguish **two kinds of updates**:
    - Proposals to update the common ground, subject to negotiation.
    - Immediate, non-negotiated updates to the common ground.
- Proposals associated with at-issue content, impositions with not-at-issue content (e.g. what's introduced by a NRRC).
- Worry: many of the features that fall out of a two-dimensional analysis need to be stipulated:
    - Non-interaction
    - Types of anchors
    - Scope of the anchor
    - Differential binding capabilities of indefinites, true quantifiers

Upgrading a dynamic semantics?

- Something you might hope for: start with a dynamic semantics, tack on a second dimension, and let the chips fall where they may.

Where we are

Data

Composition: dynamic side effects

Adding in CIs

Accounting for our data

Wrapping up

Example: nondeterminism

- It is sometimes useful to entertain multiple values in parallel:

$$\llbracket \texttt{a linguist} \rrbracket = \{x \mid \mathsf{ling}\, x\}$$
$$\llbracket \texttt{John met a linguist} \rrbracket = \{\mathsf{j}\ \mathsf{met}\ x \mid \mathsf{ling}\, x\}$$

- Usual approach is to enrich composition to handle sets:

$$\llbracket A\ B \rrbracket = \{f\,x \mid f \in \llbracket A \rrbracket \wedge x \in \llbracket B \rrbracket\}$$

- Another, equally valid possibility is to suppose that alternatives *take scope* (Charlow 2015).

Scoping alternatives

- Requires **two familiar type-shifters**.
- First: `return` is Karttunen 1977's $C_\circ$, aka Partee 1986's IDENT. It turns a boring thing into a fancy thing (though still fairly boring).

$$\texttt{return}\, x \coloneqq \{x\}$$

- Second: $\ggcurly$ turns a set m into a scope-taker by feeding each member of m to a scope κ and unioning the resulting sets.

$$m \ggcurly \kappa \coloneqq \bigcup_{x \in m} \kappa\, x$$

- E.g., $\{x \mid \text{linguist}\, x\}^{\ggcurly} = \lambda\kappa.\, \bigcup_{\text{linguist}\, x} \kappa\, x$.[1]

[1] $\{x \mid \text{linguist}\, x\}^{\ggcurly}$ is actually equivalent to the meaning Cresti 1995 assigns to *which linguist*, and also crops up in Heim 2000; Ciardelli & Roelofsen to appear.

Fancy, boring types

- Typing judgments, where $\mathtt{F}a$ should be read as "a fancy a". In this case, a fancy a is simply a set of a's, so $\mathtt{F}a ::= \{a\} ::= a \to t$:

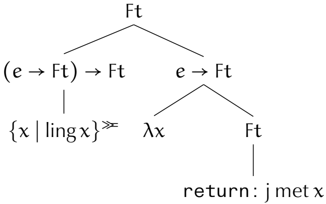
$$\mathtt{return} :: a \to \mathtt{F}a \qquad (\gg\!=) :: \mathtt{F}a \to (a \to \mathtt{F}b) \to \mathtt{F}b$$

- $\mathtt{return}$ and $\gg\!=$ build a bridge between fancy things (sets of alternatives) and boring things (familiar denotations):

$$\underbrace{m^{\gg=}}_{(a\to\mathtt{F}b)\to\mathtt{F}b}\ \overbrace{(\lambda x.\,\mathtt{return}\ \dots\ x\ \dots\ )}^{a\to\mathtt{F}b}$$

An example

- An example of how this works for *John met a linguist*:



- Gives the expected set of propositions, about different linguists:

$$\{\mathrm{j\ met}\ x \mid \mathrm{ling}\ x\}$$

- This pattern will be repeated time and again. The alternative generator takes scope via $(\ggcurly=)$, `return` applies to its remnant.

State-sensitivity: the Reader monad

- Some things (e.g., pronouns) are sensitive to the state at which they're evaluated. Suggests the following fancy type:

$$\mathsf{F}\,a ::= s \to a$$

- Along with the following monadic operations:

$$\texttt{return}\ x \coloneqq \lambda i.\, x \qquad\qquad m \gg\!= \kappa \coloneqq \lambda i.\, \kappa\,(m\, i)\, i$$

- Example, supposing $\textbf{she}_0 \coloneqq \lambda i.\, i_0$

$$\textbf{she}_0^{\gg=}\,(\lambda x.\, \texttt{return}:\ x\ \text{left}) = \lambda i.\, i_0\ \text{left}$$

Combining the two: the Reader + Set monad

- We might even combine nondeterminism and state-sensitivity, taking fancy a's to be **functions from states** into sets of a's.

$$\mathrm{F}a = s \to \{a\}$$

- This in turn implies minimally tweaked versions of `return` and $\gg\!=$:

$$\texttt{return}\,x \coloneqq \lambda i.\,\{x\} \qquad\qquad m \gg\!= \kappa \coloneqq \lambda i. \bigcup_{x \in m\,i} \kappa\,x\,i$$

- Example, supposing $\textsc{a.ling} \coloneqq \lambda i.\,\{x \mid \mathrm{ling}\,x\}$ and $\textsc{her}_0 \coloneqq \lambda i.\,\{i_0\}$.

$$\textsc{a.ling}^{\gg=}\,(\lambda x.\,\textsc{her}_0^{\gg=}\,(\lambda y.\,\texttt{return}:\ x\,\mathrm{met}\,y))$$
$$= \lambda i.\,\{x\,\mathrm{met}\,i_0 \mid \mathrm{ling}\,x\}$$

The Monad Slide

- Any `return`, $(\gg=)$ deomposes LIFT (e.g. Partee 1986):

$$(\texttt{return}\ x)^{\gg=} = \textsc{lift}\ x = \lambda\kappa.\ \kappa\, x$$

- They also form something known in category theory & computer science as a **monad** (e.g. Moggi 1989; Wadler 1992, 1995).
  - In general, monads are *really* good at allowing (arbitrarily) fancy things to interact with boring things.
  - See Shan 2002; Giorgolo & Asudeh 2012; Unger 2012; Charlow 2014 for discussions of monads in natural language semantics.

do-notation

- There's a convenient notation for working with these "LFs":

$$\begin{array}{l} \texttt{do}\ \ x \leftarrow m \\ \qquad y \leftarrow n \\ \qquad\quad \vdots \\ \qquad \texttt{return} : \phi \end{array} \quad = \ m \ggg= (\lambda x.\, n \ggg= (\lambda y.\ \cdots\ \texttt{return} : \phi\,))$$

- Standard sugaring in Haskell, essentially the "monad comprehensions" of Wadler 1992.

Examples of do-notation

- An example with alternatives (using the Set monad):

$$\begin{array}{l}\texttt{do}\;\; x \leftarrow \{x \mid \text{ling}\, x\} \\ \qquad \texttt{return}:\; \text{j met}\, x\end{array} \quad = \quad \{\text{j met}\, x \mid \text{ling}\, x\}$$

- An example with state-sensitivity (using the Reader monad):

$$\begin{array}{l}\texttt{do}\;\; x \leftarrow \lambda i.\, i_0 \\ \qquad \texttt{return}:\; \text{j met}\, x\end{array} \quad = \quad \lambda i.\, \text{j met}\, i_0$$

- And an example with both (using the Reader + Set monad):

$$\begin{array}{l}\texttt{do}\;\; x \leftarrow \lambda i.\, \{x \mid \text{ling}\, x\} \\ \qquad y \leftarrow \lambda i.\, \{i_0\} \\ \qquad \texttt{return}:\; x\, \text{met}\, y\end{array} \quad = \quad \lambda i.\, \{x\, \text{met}\, i_0 \mid \text{ling}\, x\}$$

Dynamics: basic data

- A familiar data point: Indefinites behave more like names than quantifiers with respect to anaphoric phenomena.

(10) $\{$Polly$_i$, a linguist$_i$, *every linguist$_i\}$ came in. She$_i$ sat.

Discourse referents

- Dynamic semantics: sentences add discourse referents to the "conversational scoreboard" (e.g. Groenendijk & Stokhof 1991):

$$i \longrightarrow [\![\text{Polly came in}]\!] \longrightarrow i + p$$

- Indefinites (but not quantifiers) also set up discourse referents. In case four linguists came in – a, b, c, and d – we'll have:

$$\begin{tikzcd}[row sep=small]
 & & i + a \\
 & & i + b \\
i \arrow[r] & {[\![\text{a linguist came in}]\!]} \arrow[ur] \arrow[uur] \arrow[r] \arrow[dr] & i + c \\
 & & i + d
\end{tikzcd}$$

- Formally captured by modeling meanings as relations on states. For example, here is a candidate meaning for *a linguist came in*:

$$\lambda i.\, \{i + x \mid \text{linguist}\, x \wedge \text{came}\, x\}$$

Folding in dynamics

- It's straightforward to fold dynamics into the monadic perspective.
- Dynamics relies on the ability to output modified assignments (indeed, given indefinites, to output *alternative* assignments).
- One way to think of this is in terms of a new "fancy" type:

$$\mathtt{F}a ::= s \to \{\langle a, s\rangle\}$$

- An upgrade from the previous semantics, where $\mathtt{F}a ::= s \to \{a\}$.
- The monadic operations again essentially follow from the types:

$$\texttt{return}\colon x \coloneqq \lambda i.\, \{\langle x, i\rangle\} \qquad\qquad m \gg\!= \kappa \coloneqq \lambda i. \bigcup_{\langle x,j\rangle \in m\,i} \kappa\, x\, j$$

Basic meanings

- Meaning for an indefinite:

$$\textbf{A.LING} = \lambda i.\, \{\langle x, i \rangle \mid \text{ling}\, x\}$$

- And pronouns (where i_0 is the most recently introduced dref in i):

$$\textbf{SHE}_0 = \lambda i.\, \{\langle i_0, i \rangle\}$$

Binding

- Introducing drefs for thirsty pronouns can happen modularly:

$$\mathtt{m}^{\blacktriangleright} = \mathtt{do}\ \ x \leftarrow \mathtt{m} \\ \qquad\qquad\qquad \lambda i.\,(\mathtt{return}\ x)\, i{+}x$$

- Example of how this works for an indefinite:

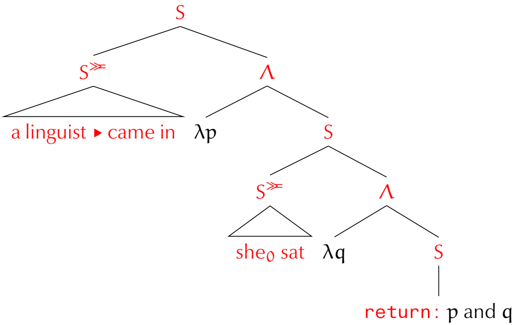
$$\textsc{a.ling}^{\blacktriangleright} = \lambda i.\, \{\langle x, i + x\rangle \mid \text{ling}\, x\}$$

- We can also ▸-shift simple type e individuals injected into the monad with `return`:

$$(\mathtt{return:}\ \mathrm{m})^{\blacktriangleright} = \lambda i.\, \{\langle \mathrm{m}, i + \mathrm{m}\rangle\}$$

Dynamic binding via LF pied-piping

- Remarkably, rejiggering the semantics in this way predicts that dynamic binding arises via a kind of "LF pied-piping":



- Unlike standard dynamic approaches, this derivation doesn't require a notion of dynamic conjunction.
  - In keeping with the approach I've been advocating, conjunction is boring and interacts with fancy things via `return` and $\gg=$.

From another angle

- The "LF" from the last slide, in terms of `do`-notation:

$$\begin{aligned}\texttt{do}\ \ & p \leftarrow \big(\texttt{do}\ \ x \leftarrow \textbf{A.LING}^{\blacktriangleright} \\ & \qquad\qquad\ \ \texttt{return}: \mathrm{left}\,x\big) \\ & q \leftarrow \big(\texttt{do}\ \ y \leftarrow \textbf{SHE}_0 \\ & \qquad\qquad\ \ \texttt{return}: \mathrm{tired}\,y\big) \\ & \texttt{return}: p \wedge q\end{aligned}$$

- The result here is equivalent to:

$$\begin{aligned}\texttt{do}\ \ & x \leftarrow \textbf{A.LING}^{\blacktriangleright} \\ & \texttt{return}: \mathrm{left}\,x \wedge \mathrm{tired}\,x\end{aligned}$$

Closure

- Important part of any dynamic semantics: operators that **quantify over alternatives**. Usually: negation, things defined in terms of it.
- Negation, type $\mathsf{F}t \to \mathsf{F}t$:

$$\textsc{not} = \lambda m.\, \lambda i.\, \{\langle \neg\exists\pi \in m\, i : \pi_0, i\rangle\}$$

- Universals, cf. $\neg\exists\neg\phi$, type $(e \to \mathsf{F}t) \to \mathsf{F}t$:

$$\textsc{every.ling} = \lambda\kappa.\, \textsc{not}\left(\begin{array}{l}\mathsf{do}\;\; x \leftarrow \textsc{a.ling} \\ \qquad\;\; \textsc{not}\,(\kappa\, x)\end{array}\right)$$

- Equivalent rendering of the universal:

$$\lambda\kappa.\, \lambda i.\, \{\langle \forall x \in \text{ling} : \exists\pi \in \kappa\, x\, i : \pi_0, i\rangle\}$$

Where we are

Data

Composition: dynamic side effects

Adding in CIs

Accounting for our data

Wrapping up

Writer: the monad for supplemental content

- Giorgolo & Asudeh 2012 point out that the **Writer** monad is useful for modeling 2-dimensional content.
- Things in the Writer monad are **pairs** of values and some supplemental content:

$$\mathsf{F}\alpha = \alpha \bullet \mathtt{t}$$

- Where $\bullet$ is just the pair constructor, i.e. $\langle \cdot\,, \cdot \rangle$. I use it to visually distinguish, and to emphasize the connection with Potts 2005.
- Injection is pairing a value with a trivial supplement. Sequencing involves *conjoining* supplemental content.

$$\texttt{return}\,x \coloneqq x \bullet \top \qquad\qquad x \bullet p \gg\!= \kappa \coloneqq v \bullet p \wedge q$$
$$\text{where } v \bullet q = \kappa\, x$$

Combining Writer with dynamics

- A fancy a can harbor nondeterminism, state-changing, and now, *supplemental content*:

$$\mathtt{F}a ::= s \to \{\langle a \bullet t, s\rangle\}$$

- Monadic operations are expressed in terms of the "underlying" dynamic monad:

$$\mathtt{return}:\ x \coloneqq \mathtt{return}:\ x \bullet \top \qquad m \gg\!= \kappa \coloneqq \mathtt{do}\ \begin{array}{l} x \bullet p \leftarrow m \\ v \bullet q \leftarrow \kappa\, x \\ \mathtt{return}:\ v \bullet p \wedge q \end{array}$$

- De-sugared version of sequencing:

$$\lambda i.\, \{\langle v \bullet p \wedge q, h\rangle \mid \langle x \bullet p, j\,\rangle \in m\, i \wedge \\ \langle v \bullet q, h\rangle \in \kappa\, x\, j\,\}$$

Where we're headed

- John, who is a linguist ($\mathsf{F}e$):

$$\texttt{return}: \mathrm{j} \bullet \mathrm{ling}\,\mathrm{j}$$

- A friend of mine, who is a linguist ($\mathsf{F}e$):

$$\begin{array}{l}\texttt{do}\ \ x \leftarrow \textbf{A.FRIEND} \\ \quad\ \texttt{return}: x \bullet \mathrm{ling}\,x\end{array}$$

- Sequencing any of these $\mathsf{F}e$'s with the rest of the sentence:

$$\begin{array}{l}\texttt{do}\ \ x \leftarrow m \\ \quad\ \texttt{return}: \mathrm{left}\,x\end{array}$$

- No problem for supplemental things to interact with boring things.

Dynamic interactions

- Key bit: Writer + Dynamic monad is still a State monad!
- This means it is *totally kosher* to sequence something in the dynamic monad with a WriterT dynamic program:

$$\begin{array}{l} \texttt{do}\ \ x \leftarrow m \\ \qquad \texttt{do}\ \ \ldots\ldots \quad \checkmark \end{array}$$

- A trivial but revealing example – lifting a $\mathsf{F}e$ into a $\mathsf{F}e$:

$$\begin{array}{l} \texttt{do}\ \ x \leftarrow \textbf{A.LING}^{\blacktriangleright} \\ \qquad \texttt{return}\ x \end{array} \quad = \quad \lambda i.\, \{\langle x \bullet \top, i + x\rangle \mid \mathrm{ling}\, x\}$$

- Indeed, *any* $\mathsf{F}a$ can be turned into a $\mathsf{F}a$, and any a into an $\mathsf{F}a$.

Comma

- Semantics for the comma intonation – turns a restrictive relative clause into a non-restrictive relative.

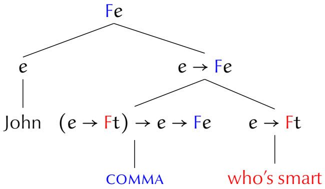
$$\textbf{COMMA}\ \kappa = \lambda x.\ \texttt{do}\ \begin{array}{l} p \leftarrow \kappa\, x \\ \texttt{return:}\ x \bullet p \end{array}$$

- Type: $(e \to Ft) \to e \to Fe$
- De-sugared (notice that the type is $e \to Fe$):

$$\lambda x.\, \lambda i.\, \{\langle x \bullet p, j\rangle \mid \langle p, j\rangle \in \kappa\, x\, i\}$$

Basic example

- A structure for *John, who's smart*:[2]



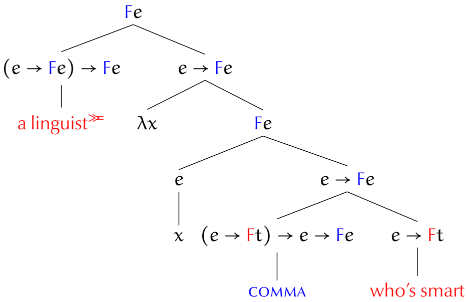
- Composing up via simple functional application, we end up with the following, as expected:

$$\begin{aligned} \textbf{COMMA}\,(\lambda x.\, \texttt{return}\colon \mathrm{smart}\, x)\, j &= \lambda i.\, \{\langle j \bullet \mathrm{smart}\, j, i\rangle\} \\ &= \texttt{return}\colon j \bullet \mathrm{smart}\, j \end{aligned}$$

[2] Suppressing derivation of the relative clause, but see Charlow 2014 for details.

Bridging the monads

- Can now fold in an indefinite, similarly:



- Yields the following meaning (again, composing by simple FA):

$$\begin{aligned}&\texttt{do}\;\; x \leftarrow \textbf{A.LING}^{\blacktriangleright}\\ &\qquad \textbf{COMMA}\,(\lambda x.\, \texttt{return}:\ \text{smart}\, x)\, x\end{aligned}$$

- Bottles up a nondeterministic value, supplement, updated state:

$$= \lambda i.\, \{\langle x \bullet \text{smart}\, x, i{+}x\rangle \mid \text{ling}\, x\}$$

Where we are

Data

Composition: dynamic side effects

Adding in CIs

Accounting for our data

Wrapping up

Independence

- Why do universals, negation, etc. appear not to interact with appositive content?
- Their types are incompatible! E.g. universal needs to take scope over a $(e \to \mathsf{F}t) \to \mathsf{F}t$! It can't do anything with an $e \to \mathsf{F}t$.
- For exactly the same reason, it is impossible to anchor a NRRC to a universal. The types just don't fit.
- More generally, it appears that we don't require any meanings with negative occurrences of $\mathsf{F}\alpha$ types (save for "grammatical" operations like `return`, $\ggcurly$, and perhaps λ).
  - This very closely mirrors the situation in Potts 2005.

Scope of the anchor

- Works for the same reason.
- Type of e.g. negation: $F\mathtt{t} \to F\mathtt{t}$.
- There is just no way to combine this with an $F\mathtt{t}$. The negation doesn't know what to do with the extra dimension of content!
- But that's the only way for negation to scope over an indefinite-anchored NRRC! And that just won't work.

Binding into a NRRC

- A linguist knows John, who likes her:

$$\begin{array}{l} \texttt{do}\ \ x \leftarrow \textsf{A.LING}^{\blacktriangleright} \\ \qquad \texttt{do}\ \ y \leftarrow \lambda i.\, \{\langle j \bullet \text{j likes } i_0, i\rangle\} \\ \qquad\qquad \texttt{return}:\ x \text{ knows } y \end{array}$$

- Evaluated and de-sugared:

$$\lambda i.\, \{\langle x \text{ knows j} \bullet \text{j likes } x, i + x\rangle \mid \text{ling}\, x\}$$

- The important bit: $\textsf{A.LING}^{\blacktriangleright}$ can bind into the appositive because *it makes sense* to sequence a $F\alpha$ with a $F\alpha$ (here, Fe and Ft).
- Again, the reason is that any $F\alpha$ is a $F\alpha \bullet t$.

Binding out of a NRRC

- John, who knows a linguist, likes her:

$$\begin{aligned}&\mathtt{do}\ \ x \leftarrow \lambda i.\,\{\langle j \bullet j \text{ knows } v, i + v\rangle \mid \text{ling}\, v\}\\ &\qquad y \leftarrow \left(\mathtt{do}\ \ z \leftarrow \textbf{HER}_0\right.\\ &\qquad\qquad\qquad \left.\mathtt{return}:\ z\right)\\ &\qquad \mathtt{return}:\ x \text{ likes } y\end{aligned}$$

- Evaluated and de-sugared:

$$\lambda i.\,\{\langle j \text{ likes } v \bullet j \text{ knows } v, i + v\rangle \mid \text{ling}\, v\}$$

No binding for true quantifiers

- For, e.g., a universal to bind **into** an appositive, the universal would need to **take scope over** the appositive.
- Again, their types are incompatible. A universal needs to take scope over a $(e \to \mathsf{F}t) \to \mathsf{F}t$! It can't do anything with an $e \to \mathsf{F}t$.
- For, e.g., a universal to bind **out of** an appositive, you'd need the universal to be externally dynamic. Which it isn't:

$$\textsc{every.ling} = \lambda\kappa.\,\textsc{not}\left(\begin{array}{l}\mathtt{do}\;\; x \leftarrow \textsc{a.ling} \\ \qquad\;\; \textsc{not}\,(\kappa\, x)\end{array}\right)$$

Exceptional scope

- Since the theory I've proposed relies on **scope-taking**, you might expect that we have to appeal to exceptional scope-taking to explain cases when an NRRC occurs in an island.
- Well, yes and no. We do, but exceptional scope actually just *falls out of the monadic approach*. See Charlow 2014, 2015 for details.

Where we are

Data

Composition: dynamic side effects

Adding in CIs

Accounting for our data

Wrapping up

Wrapping up

- Super natural to enrich the dynamic monad (nondeterminism and state) with supplemental content.
  - Involves nothing but the WriterT transform and a semantics for the comma intonation!
- Predicts a number of properties of NRRCs:
  - Independence/non-interaction
  - Which things can bind into and out of appositives, which can't
  - Scope of the anchor
- Fully compositional. No need for a representation language — direct model-theoretic interpretation.
  - Everything happens via functional application, with monadic combinators greasing the compositional skids.

References

AnderBois, Scott, Adrian Brasoveanu & Robert Henderson. 2015. At-issue Proposals and Appositive Impositions in Discourse. *Journal of Semantics* 32(1). 93–138.

Charlow, Simon. 2014. *On the semantics of exceptional scope*: New York University Ph.D. thesis.

Charlow, Simon. 2015. The scope of alternatives. Talk presented at SALT 25.

Ciardelli, Ivano & Floris Roelofsen. to appear. Alternatives in Montague Grammar. In *Proceedings of Sinn und Bedeutung 19*, xx–xx.

Cresti, Diana. 1995. Extraction and reconstruction. *Natural Language Semantics* 3(1). 79–122.

Giorgolo, Gianluca & Ash Asudeh. 2012. $\langle M, \eta, \star \rangle$: Monads for conventional implicatures. In Ana Aguilar Guevara, Anna Chernilovskaya & Rick Nouwen (eds.), *Proceedings of Sinn und Bedeutung 16*, 265–278. MIT Working Papers in Linguistics.

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

Heim, Irene. 2000. Notes on Interrogative Semantics. Unpublished lecture notes.

Karttunen, Lauri. 1977. Syntax and semantics of questions. *Linguistics and Philosophy* 1(1). 3–44.

Moggi, Eugenio. 1989. Computational lambda-calculus and monads. In *Proceedings of the Fourth Annual Symposium on Logic in computer science*, 14–23. Piscataway, NJ, USA: IEEE Press.

Partee, Barbara H. 1986. Noun Phrase Interpretation and Type-shifting Principles. In Jeroen Groenendijk, Dick de Jongh & Martin Stokhof (eds.), *Studies in Discourse Representation Theory and the Theory of Generalized Quantifiers*, 115–143. Dordrecht: Foris.

References (cont.)

Potts, Christopher. 2005. *The logic of conventional implicatures*. Oxford: Oxford University Press.

Shan, Chung-chieh. 2002. Monads for natural language semantics. In Kristina Striegnitz (ed.), *Proceedings of the ESSLLI 2001 Student Session*, 285–298.

Unger, Christina. 2012. Dynamic Semantics as Monadic Computation. In Manabu Okumura, Daisuke Bekki & Ken Satoh (eds.), *New Frontiers in Artificial Intelligence JSAI-isAI 2011*, vol. 7258 Lecture Notes in Artificial Intelligence, 68–81. Springer Berlin Heidelberg.

Wadler, Philip. 1992. Comprehending monads. In *Mathematical Structures in Computer Science*, vol. 2 (special issue of selected papers from 6th Conference on Lisp and Functional Programming), 461–493.

Wadler, Philip. 1995. Monads for functional programming. In Johan Jeuring & Erik Meijer (eds.), *Advanced Functional Programming*, vol. 925 Lecture Notes in Computer Science, 24–52. Springer Berlin Heidelberg.