Binding in alternative semantics

February 25, 2015

1 Today

- Thusfar, we've been a bit coy about how alternatives interact with predicate abstraction.
- I've noted at a couple points that there really is no "true" abstraction operation available in alternative semantics. More specifically, adopting a point-wise approach to composition turns out to be incompatible with standard approaches to functional abstraction.
- Today, we'll explore this in some detail. We'll look at a few ideas about how abstraction might be implemented in an alternative semantics. And we'll see why each of them fails. Tellingly, the issues we identify will turn out to mirror some of the problems we identified for choice functions.

2 Predicate abstraction refresher

- What's predicate abstraction good for? The answer is somewhat dependent on the framework you work with. For Shan (2004) (following Jacobson 1999), the answer is nothing! The rest of us mortals will need predicate abstraction to deal, at least, with:
  - ▷ Binding: e.g. *no student$_i$ completed their$_i$ homework*. PA is used to bind the variable pronoun *their*, i.e. to mess with the assignment function in such a way as to guarantee that *their* and the subject are covalued.
  - ▷ Scope displacement: e.g. *a guard stood in front of every embassy*. QR takes the object-position quantifier out of its in situ position, moves it to one of sentential scope. A trace is left behind, and abstracted over to create a property that serves as the QR'd quantifier's argument.
- Predicate abstraction, formally:

$$[\![n\ X]\!]^g \coloneqq \lambda x.\, [\![X]\!]^{g[x/n]}$$

  As in the semantics of the λ-calculus, an *abstraction index* triggers a syncategorematic interpretation rule. The function of this rule is to introduce a new functional argument, and then to make sure that any pronouns or traces within the scope of, and co-indexed with, the abstraction index evaluate to that argument.

- A basic example:

$$\begin{aligned}[\![3\ [t_3\ \text{left}]]\!]^g &= \lambda x.\, [\![t_3\ \text{left}]\!]^{g[x/3]}\\ &= \lambda x.\, \mathsf{left}(x)\end{aligned}$$

3 Alternative semantics

- A refresher on the standard way to lift a semantics into one with alternatives (Hamblin 1973; Rooth 1985, 1992). Meanings of type α systematically replaced with meanings of type $\alpha \to t$, and a new semantics for binary composition:

$$[\![X\ Y]\!] = \{x(y) : x \in [\![X]\!] \wedge y \in [\![Y]\!]\}$$

- Predicate modification can be lifted in a similar way:

$$[\![X\ Y]\!] = \{x \cap y : x \in [\![X]\!] \wedge y \in [\![Y]\!]\}$$

- More generally, for two-place function f, a *point-wise* version f' can be defined, as follows (i.e. previously we instantiated f as functional application):

$$f'(X)(Y) \coloneqq \{f(x)(y) : x \in X \wedge y \in Y\}$$

- How about predicate abstraction? Is there a simple abstraction rule, along the lines of application/modification? Turns out, the answer is negative (Rooth 1985; Shan 2004; Romero & Novel 2013; Charlow 2014).
- Let's see what happens when we apply the PA rule from before (Romero & Novel 2013 term this "naive" predicate abstraction):

$$\begin{aligned}[\![3\ [_{v\mathrm{P}}\ t_3\ \text{met a linguist}]]\!]^g &= \lambda x.\, [\![v\mathrm{P}]\!]^{g[x/3]}\\ &= \lambda x.\, \{\mathsf{met}(x,\, y) : \mathsf{ling}(y)\}\end{aligned}$$

- In a sense, this gives completely reasonable results. The sister of the abstraction index denotes something of type $t \to t$ (a set of "propositions"), which PA maps into something of type $e \to t \to t$. Naive PA is a "true" abstraction operation.

- The problem comes in when we try to make use of these meanings. For example, we should expect a post-abstraction denotation in cases like this to be the sort of thing that combines with a quantifier. In alternative semantics, the meaning of e.g. *nobody* is the following, i.e. a singleton set containing the usual generalized quantifier meaning and nothing else:

$$\{\textsf{nobody}\}$$

- This meaning has type $((e \to t) \to t) \to t$. The meaning derived by "naive" PA has type $e \to t \to t$. These two functions cannot combine, either by regular or point-wise functional application.

4 Hamblinized abstraction

- Diagnosis: to get the types to work out, we need the output of abstraction to be a *set of functions*, rather than a *function into sets*. The former of these, but not the latter, can compose with a generalized quantifier via point-wise functional application.

- In particular, we would like to find a rule for abstraction on which the following equivalence holds:

$$[\![3\ [t_3 \text{ met a linguist}]]\!]^g = \{\lambda x.\, \textsf{met}(x,\ y) : \textsf{ling}(y)\}$$

- A rule for abstraction with the correct types has in fact been proposed (e.g. Hagstrom 1998; Kratzer & Shimoyama 2002):

$$[\![n\ X]\!]^g := \{f : \forall x.\, f(x) \in [\![X]\!]^{g[x/n]}\}$$

- Yet while this rule is certainly of the correct type, does it deliver the correct set of functions — that is, would it validate the equivalence above?

- Alas, no. It turns out that, when the sister of the abstractor index denotes a *singleton* set, PPA yields essentially the same result as regular PA:

$$\begin{aligned} [\![3\ [_{v\text{P}}\ t_3 \text{ met Bill}]]\!]^g &= \{f : \forall x.\, f(x) \in [\![v\text{P}]\!]^{g[x/3]}\} \\ &= \{f : \forall x.\, f(x) \in \{\textsf{met}(x,\ \textsf{b})\}\} \\ &= \{f : \forall x.\, f(x) = \textsf{met}(x,\ \textsf{b})\} \\ &= \{\lambda x.\, \textsf{met}(x,\ \textsf{b})\} \end{aligned}$$

- But we get into trouble when the sister of the abstraction denotes a non-singleton set of alternatives, as in a configuration like the following:

$$[_\Lambda\ 3\ [_{v\text{P}}\ t_3\ [_{\text{VP}} \text{ met } [_{\text{DP}} \text{ a linguist}]]]$$

- Going node-by-node, we find that everything proceeds as expected, with the alternatives simply expanding up the tree, until we hit Λ:

$$\begin{aligned} [\![\text{DP}]\!]^g &= \{y : \textsf{ling}(y)\} \\ [\![\text{VP}]\!]^g &= \{\lambda x.\, \textsf{met}(x,\ y) : \textsf{ling}(y)\} \\ [\![v\text{P}]\!]^g &= \{\textsf{met}(g(3),\ y) : \textsf{ling}(y)\} \\ [\![\Lambda]\!]^g &= \{f : \forall x.\, f(x) \in [\![v\text{P}]\!]^{g[x/3]}\} \\ &= \{f : \forall x.\, f(x) \in \{\textsf{met}(x,\ y) : \textsf{ling}(y)\}\} \end{aligned}$$

- This formula is somewhat complicated to think about. How does it compare to the formula we were after, i.e. one naming a set of functions $\lambda x.\, \textsf{met}(x,\ y)$, for y some linguist?

- Well, it certainly includes all of those functions. But the issue is that it potentially includes many more. In words, if **A** is the set we would like to have derived, and **B** is the set of functions we have actually derived, there will be models in which:

$$\mathbf{A} \subset \mathbf{B}$$

- For instance, here is a function that could be in **B** but not in **A**:

$$\left\{\begin{array}{c} x_1 \mapsto \textsf{met}(x_1,\ l_1) \\ x_2 \mapsto \textsf{met}(x_2,\ l_{35}) \\ x_3 \mapsto \textsf{met}(x_3,\ l_{894}) \\ \vdots \end{array}\right\}$$

  This function is "about" different linguists, in a way no function in **A** is. Moreover, there will be *many* such "mixed" functions in **B** (of course, subject to the richness of our model).

- In a nutshell, then: the problem with this abstraction rule is that it does not (cannot) guarantee that the functions it returns are **uniform** with respect to the linguists. What we need are functions which only implicate one linguist, which this abstraction rule does not provide us.

- The problem will only get more acute as we start larding up the abstracted-over constituent with alternative generators.
- In fact, Kratzer & Shimoyama 2002 are aware that the result here is not quite correct, i.e. that is "does not quite deliver the expected set of functions". However, they indicate that they are not aware of any incorrect predictions made by these extra functions.
- A pressing question, then: *are* there any?
- Satarupa will illustrate some potential issues with this and one possible solution. Then we will see some others. The problem will turn out to be quite sticky.

5 The wages of in situ

- Thanks Satarupa! I want to discuss Problems 1 and 2 in some more detail, in particular in a way that connects those issues with our discussion of choice functions.
- First, Problem 1 is a too-many-functions problem. That is, the abstraction rule given above is too promiscuous; e.g. in the case above, it includes functions that are "about" different linguists, in addition to the correct functions — the functions which are each about precisely one linguist.
- As Satarupa, following Shan 2004 showed, this leads to some odd predictions if we're using Hamblin semantics for questions.
- Consider now how an alternative semantics, with the too-many-functions abstraction rule, would handle a sentence like the following:

$$\text{nobody } [_{\Lambda}\ 3\ t_3 \text{ met a linguist}]$$

- As we've seen, we derive the following meaning for this sentence's Λ. And as we discussed, this set includes "mixed" functions in addition to "uniform" ones.

$$\{f : \forall x.\, f(x) \in \{\mathsf{met}(x,\ y) : \mathsf{ling}(y)\}\}$$

- Assume that for any x, x didn't meet some linguist l_x. Importantly, the above set includes the following function — in prose, a function which maps any individual x to the proposition that x met a certain linguist x *didn't* meet:

$$\lambda x.\, \mathsf{met}(x,\ l_x)$$

- Of course, nobody has that property. Thus, we incorrectly predict that *nobody met a linguist* can be true, even in a model where everybody met every linguist but one (i.e. for any x, l_x). That is, the sentence ends up with the same truth conditions as (the surface-scope reading of) *nobody met every linguist*.
- A similar issue arises vis a vis Problem 2.
- As Romero & Novel (2013) (following Rooth 1985; Poesio 1996) point out, the abstraction issue in a sense arises from alternative semantics being formulated in a certain way, when it could be formulated otherwise.

$$[\![X]\!] = \lambda g.\, \{\cdots\} \qquad [\![X]\!] = \{\lambda g.\, \cdots\}$$

- Moving assignment functions into the model has a lovely consequence. We can define a fully compositional version of the abstraction operator (e.g. Kobele 2010):

$$\lambda_n \coloneqq \lambda p.\, \lambda g.\, \lambda x.\, p(g[x/n])$$

- Now, it is trivial to define a Hamblinized version of this function, which composes point-wise with another set to yield a set of functions:[1]

$$[\![n]\!] \coloneqq \{\lambda_n\}$$

- This sort of approach has quite good coverage. Near as I can tell, it's useful for dealing with alternatives when we're interested in calculating focus values, and implicatures.
- However, it runs into trouble in other domains, specifically when the alternative generator is the sort of thing that can be *restricted*.
- That is, we seem to have replaced our too-many-functions problem with a how-many-functions problem.

[1]This is one of those solutions that keeps being re-discovered. It was essentially given in Rooth 1985, though this seems to have been neglected in the recent literature. Ede Zimmermann informs me that he is responsible for its presence in Poesio 1996.

- Romero & Novel (2013) suggest treating certain alternative generators as indeterminate definite descriptions whose values range over the entire domain of individuals. That is, they answer the how-many-functions question by including every possible relevant function.

- Another possibility is to assimilate the indefinite determiner to a choice function, which (to a first approximation) will give the same results:

  $$[\![\text{a paper he}_3 \text{ wrote}]\!] = \{\lambda g.\, f(\{y : y \text{ is a paper } g(3) \text{ wrote}\}) : f \in \mathsf{CH}\}$$

- Crucially, however, both of these approaches will predict that an indefinite can out-scope something that binds into its restrictor, something we have already seen is impossible with reference to cases such as *no candidate*$_i$ *submitted a paper she*$_i$ *wrote* (Schwarz 2001).

6 Taking stock

- Ciardelli & Roelofsen (to appear) give an alternative semantics that uses naive predicate abstraction. So for example the result of abstracting over a propositional node will (as before) have type $e \to t \to t$.

- They solve the issue of integrating these meanings by lifting the types of quantifiers to $(e \to (t \to t)) \to (t \to t)$. This allows the output of naive predicate abstraction to compose with quantifiers directly by functional application to give a set of propositions, type $t \to t$. We have and eat our cake.

- This is, I would argue, not actually a solution.

- Consider: there is not actually any mystery about how to have the meanings we end up with be sets. Karttunen (1977) (and then Heim 2000) showed how it could be done using, essentially, nothing besides bare-bones functional application.

- Indeed, Heim 2000 (by way of extending the Karttunen semantics to indeterminate pronouns) showed how the general strategy could work to derive alternative sets of any type, i.e. not just sets of propositions.

- However, in those accounts, the question of *island-hood* is not addressed. That is, to end up with sets of alternatives, something needs to *take scope* in a certain way. Concretely for questions, the *wh* word needs to be moved (either overtly or covertly) above C°.

- Though we did not explore how a Karttunen semantics might work for regular indefinites, something similar would hold there. We would have alternative sets, but deriving them would require the indefinite to take scope above something analogous to C°.

- As we saw last week, we could in principle combine alternatives with choice functions, but this will bring in all the problems associated with choice functions.

- In other words, the difficulty is not getting your semantics to derive alternative sets. Rather, the difficulty is deriving alternative sets in a way that uses alternatives to answer hard questions about the syntax-semantics interface. This *can* be done, and in a conservative way. That is where we're headed.

References

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

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

Hagstrom, Paul. 1998. *Decomposing Questions*: Massachusetts Institute of Technology Ph.D. thesis.

Hamblin, C. L. 1973. Questions in Montague English. *Foundations of Language* 10(1). 41–53.

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

Jacobson, Pauline. 1999. Towards a Variable-Free Semantics. *Linguistics and Philosophy* 22. 117–184. doi:10.1023/A:1005464228727.

Karttunen, Lauri. 1977. Syntax and semantics of questions. *Linguistics and Philosophy* 1(1). 3–44. doi:10.1007/BF00351935.

Kobele, Gregory M. 2010. Inverse linking via function composition. *Natural Language Semantics* 18(2). 183–196. doi:10.1007/s11050-009-9053-7.

Kratzer, Angelika & Junko Shimoyama. 2002. Indeterminate Pronouns: The View from Japanese. In Yukio Otsu (ed.), *Proceedings of the Third Tokyo Conference on Psycholinguistics*, 1–25. Tokyo: Hituzi Syobo.

Poesio, Massimo. 1996. Semantic Ambiguity and Perceived Ambiguity. In Kees van Deemter & Stanley Peters (eds.), *Semantic Ambiguity and Underspecification* (CSLI Lecture Notes 55), 159–201. Stanford: CSLI Publications.

Romero, Maribel & Marc Novel. 2013. Variable Binding and Sets of Alternatives. In Anamaria Fălăuș (ed.), *Alternatives in Semantics*, chap. 7, 174–208. Houndsmills, Basingstoke, Hampshire: Palgrave Macmillan.

Rooth, Mats. 1985. *Association with Focus*: University of Massachusetts, Amherst Ph.D. thesis.

Rooth, Mats. 1992. A theory of focus interpretation. *Natural Language Semantics* 1(1). 75–116. doi:10.1007/BF02342617.

Schwarz, Bernhard. 2001. Two kinds of long-distance indefinites. In Robert van Rooy & Martin Stokhof (eds.), *Proceedings of the Thirteenth Amsterdam Colloquium*, 192–197. University of Amsterdam.

Shan, Chung-chieh. 2004. Binding alongside Hamblin alternatives calls for variable-free semantics. In Kazuha Watanabe & Robert B. Young (eds.), *Proceedings of Semantics and Linguistic Theory 14*, 289–304. Ithaca, NY: Cornell University.