Sid Su

12-18-21

PHIL 478

The Problem of Pronouns - Variable-Free Semantics

Background

Language can be modeled using “types”. *e* is the type of objects, *t* is the type of sentences and these two types can be combined together to form new types, for instance $e \to t$ is a type of objects to sentences. Language is modeled by representing each part of speech as a type, and deriving meaning by applying “rules" to the types. The simplest and most important rule says that a type from $a \to b$ can be combined with an input a to form type b. This rule can be be represented as an upside down proof tree:

$$\frac{f\,x : b}{f : a \to b \qquad x : a}$$

For example, in the sentence

(1) Amanda saw Bella

Proper nouns are of type *e*, “Amanda” and “Bella”, and transitive verbs are of type $e \to e \to t$, “saw”. When $e \to e \to t$ (saw) combines with *e* (Bella), they form the type of intransitive verb $e \to t$, (saw Bella), which then combines with another type *e* (Amy) to form, *t,* the type of sentences “Amanda saw Bella”. Or as an upside down proof tree:

$$\dfrac{\mathrm{a} : \mathtt{e} \qquad \dfrac{\mathrm{saw} : \mathtt{e} \to \mathtt{e} \to \mathtt{t} \qquad \mathrm{b} : \mathtt{e}}{\mathrm{sawb} : \mathtt{e} \to \mathtt{t}}}{\mathrm{sawba} : \mathtt{t}}$$

From the upside down proof tree, we see that the parts of speech can be represented as relations and objects in predicate logic, but the scope of this paper will focus on the logic of types, since that's what is relevant when considering pronouns.

Introduction

I will begin by introducing the "standard account" of language type-semantics. Then I will present the problem with pronouns when doing type-semantics. Then I will present an alternate theory, the so-called "variable-free semantics". I will conclude with some remarks about the limitations of the theory.

Pronouns in the "Standard Account"

Since pronouns reference other objects, when considering them in type semantics we add subscripts for what they refer to, and the pronoun being used:

(2) Amy_i thought that Amanda saw her_i.

In this case, the "her_i" refers to "Amy_i", because the subscripts match.

In the Standard Account pronouns are of type e^g. *e* is the type of object, and the superscript *g* represents an assignment function to another object. This seems intuitive for pronouns because a pronoun seems to just be placeholders for a proper noun, however, the type e^g leads to a *type mismatch* when trying to evaluate sentences with types. It is not possible to apply e^g to anything, because they expect e, not type e^g. For instance, if we wanted to apply e^g to the intransitive verb of type $e \rightarrow t$, we couldn't, because e^g does not go into e.

The solution with the Standard Account is to have another assignment function for the intransitive verb so the types match, so an intransitive verb would be of type $(e \rightarrow t)^g$. However, this is unideal because once any pronouns are used in a sentence, every other part of the sentence needs an assignment function in order to avoid type mismatch. Much of the time the pronoun really is just referring to another part of a sentence, and makes up a small part of the sentence, it doesn't seem reasonable to change every part of the sentence in order to conform with the pronoun, I will refer to this problem as *assignment function corruption*.

Variable-Free Theory

The variable-free theory, running contrary to the Standard Account, tries to remedy this by avoiding assignment functions altogether. Instead, variable-free semantics:

1) Adding a type for pronouns e^e
2) Adding rules to carry and resolve pronouns.

Type Shifting (Lift)

Proper nouns can be of type e or $(e \rightarrow t) \rightarrow t$. Having a more flexible type of proper nouns is useful because in speech they can occupy the role of a subject or an object. In the case of a subject, proper nouns receive intransitive verb phrases of type $e \rightarrow t$ to output a sentence of type t, while as an object proper nouns combine with transitive verbs $e \rightarrow e \rightarrow t$ to become intransitive verb phrases of type $e \rightarrow t$.

In variable-free semantics, the type of pronouns becomes $e \to e$, the type of objects to other objects, which is also the identity function. For the sake of clarity this is usually denoted e^e, but it's important to note that it cannot be used directly as a substitute for *e*.

'G' and 'Z' Rules

Only changing the type of pronouns is insufficient to completely resolve the issue of type mismatch, since $e \to e$ or e^e still does not interact with other types well. In order to resolve the pronoun type, new rules must be added.

The 'G' rule is used in the case where pronoun meaning is still ambiguous and has not yet been resolved. It says that $a \to b$ can combine with type a^c to form type b^c. This is more formally called "passing up anaphoric dependencies", because it's leaving the pronoun's anaphora, what it refers to, out in the open and to be closed by the Z rule. The G rule can be represented as an upside down proof tree as follows:

$$\frac{\lambda_x f\,(m\,x) : \boldsymbol{b}^c}{f : \boldsymbol{a} \to \boldsymbol{b} \qquad m : \boldsymbol{a}^c}\ \mathsf{G}$$

Consider the sentence:

(3) Robert saw him.

"Robert" is a proper noun of type $(e \to t) \to t$, "saw" is a transitive verb of type $e \to e \to t$ and "him" is a pronoun of type e^e. Apply the G rule to combine "saw" ($e \to e \to t$) with "him" (e) to form the intransitive verb phrase "saw him" $(e \to t)^e$. Because what "him" is referring to is still unknown, the superscript *e* is still carried. The G rule is applied again, and the proper noun "Robert" $((e \to t) \to t)$ is combined with the intransitive verb phrase "saw him" ($(e \to t)^e$) to form the sentence "Robert saw him" of type t^e. Note that even sentences can have anaphoric dependencies. We still do not know who "him" is, which is why the sentence is of type t^e, and it is up to the context of the situation or a conjunction with another sentence to fill in the gaps of the extra lambda term provided by type t^e.

The 'Z' rule is used to resolve a pronoun, and is applied the step right before the pronoun is merged with what it is referring to. It says that type *a→c→b* can combine with type a^c to form *c→b*. This is more formally called closing anaphoric dependencies. The Z rule can be represented as an upside down proof tree as follows:

$$\frac{\lambda_x f\,(m\,x)\,x : c \to b}{f : a \to c \to b \qquad m : a^c}\ \mathsf{Z}$$

Consider the sentence:

(4) Brian_i thought Robert saw him_i.

"Brian" is a proper noun of type *(e→t)→t*, "thought" is a transitive verb in using the alternate type *t→e→t* and from example (3), "Robert saw him" has the type t^e. The Z rule is applied, because $\boldsymbol{t^e}$ matches up with ***t→e→****t*, and we know from subscripts that Brian is who "him" is referring to, yielding the intransitive verb phrase "thought Robert saw him" *(e→t)*. This combines normally with Brian *((e→t)→t)* to end up with a sentence *t*. A full upside down proof tree can be made for the whole derivation:

$$\frac{\mathrm{eb}\,(\lambda_x\,\mathrm{thinks}\,(\mathrm{saw}\,x\,\mathrm{a})\,x) : \mathbf{t}}{\mathrm{eb} : (\mathbf{e} \to \mathbf{t}) \to \mathbf{t} \qquad \dfrac{\lambda_x\,\mathrm{thinks}\,(\mathrm{saw}\,x\,\mathrm{a})\,x : \mathbf{e} \to \mathbf{t}}{\mathrm{thinks} : \mathbf{t} \to \mathbf{e} \to \mathbf{t} \qquad \dfrac{\lambda_x\,\mathrm{saw}\,x\,\mathrm{a} : \mathbf{t}^{\mathbf{e}}}{\dfrac{\lambda_\kappa\,\kappa\,\mathrm{a} : (\mathbf{e} \to \mathbf{t}) \to \mathbf{t}}{\mathrm{a} : \mathbf{e}}\,\mathsf{Lift} \qquad \dfrac{\lambda_x\,\mathrm{saw}\,x : (\mathbf{e} \to \mathbf{t})^{\mathbf{e}}}{\mathrm{saw} : \mathbf{e} \to \mathbf{e} \to \mathbf{t} \qquad \lambda_x\,x : \mathbf{e}^{\mathbf{e}}}\,\mathsf{G}}\,\mathsf{G}}\,\mathsf{Z}}$$

Lingering Issues with Variable Free Semantics

There are still problems with variable free semantics. One such problem is that it still has trouble deciphering ambiguous language. Consider a sentence similar to the previous one:

(5) Every boy thought Amanda saw him.

The meaning of "him" in this sentence is ambiguous, because it could be referring to *Every Boy*, but it could also be referring to someone else Amanda saw, for instance if Amanda saw a famous actor walking on the street the other day. Knowing which rule to apply, G or Z, without subscripts still depends on an "intuitive" understanding of what the speaker is saying, that is to say we cannot do blind surface interpretation of the sentence, because in order to resolve it we need to do either G or Z rules. This is inbuilt into variable-free semantics, because G and Z rules are applied before the next operation knows if the object is what the pronoun is referring to. We are committed to keeping open or closing the anaphoric dependencies before knowing what would potentially close it.

Another problem is the application of G and Z rules when there are multiple pronouns. Consider:

(6) He_i saw her_j leave him_k.

It's unclear how to apply G and Z rules for multiple pronouns, with some solutions distributing anaphoric dependencies to each broken down part of the type.

Conclusion

Variable-Free semantics provides an approach to pronouns when modeling language with types. It does this by changing the type of pronouns and adding rules for how to combine other types with pronouns. Though it is not without flaws, and there are some issues that have yet to be worked out.