DRT and Situation Semantics

Paper for Ling237, Semantics Seminar on Situation Semantics (Spring 2006)

Iddo Lev

April 16, 2007

1 Overview

In a context where John and Mary are running against each other in an election, the salient reading of the sentence:

(1) John and Mary think they will win.

is that John thinks he will win and Mary thinks she will win.

In DRT (Kamp, 1981; Kamp and Reyle, 1993; van Eijck and Kamp, 1997), and its compositional λ-DRT versions such as (Bos et al., 1994), this sentence poses a problem. To sketch the problem, the issue is that the compositional construction will create something like:

(2) $[z = john \oplus mary \mid (\lambda x.think(x, [y_{they} =? \mid win(y)]))^{\Box}(z)]$

The $^{\Box}$ here is my addition. It signals that the formula is underspecified regarding whether the predicate is applied on z collectively or distributively.

The problem here is that the discourse variable y needs to be resolved to x. However, y and x are different kinds of terms. The former is a discourse variable while the latter is a normal variable.

Equating the two variables poses the following problem. In the interpretation of DRSs using a model-theoretic semantics (along the lines of (Muskens, 1995, 1996; Kohlhase et al., 1996; van Eijck and Kamp, 1997)), discourse variables denote entities in the domain. In contrast, a lambda-abstracted variable like x above does not denote anything. It is just a notational convenience. There is even a way to get rid of such variables by using only combinatory functors (see e.g. (Carpenter, 1998, section 2.5)).[1] No such omission is possible with discourse variables.

In (Kamp and Reyle, 1993), a similar sentence with a distributive reading specifies the $^{\Box}$ in terms of a duplex condition:

(3) $[z = john \oplus mary \mid$
$\quad [u \mid u \in z] \Rightarrow [\ \mid (\lambda x.think(x, [y =? \mid win(y)]))(u)]\ ] \quad \equiv$
$[z = john \oplus mary \mid$
$\quad [u \mid u \in z] \Rightarrow [\ \mid think(u, [y =? \mid win(y)])]\ ]$

[1] A very simple illustration of this point is that a composition of two functions f and g need not be written as $\lambda x.f(g(x))$ as it can be written using the functor $f \circ g$ without resorting to an "intermediate" variable x.

Here the discourse variable y can be equated with the discourse variable u to get the correct truth conditions.

There is a feeling that this solution is not satisfactory because the discourse variable u is not independently motivated, and y should really be resolved to x. Furthermore, we would like to have a form that retains the underspecified $\Box$ but resolves the y. Crucially, this solution will not work for a sentence such as:

(4) John and Mary think they like each other.

with the desired representation:

(5) $[z = john \oplus mary \mid \textsc{recip}(z, (\lambda x \lambda u.think(x, [y =? \mid like(y, u)])))]$

To obtain the desired reading, where John thinks he likes Mary and Mary thinks she likes John, the discourse variable y needs to be resolved to the normal variable x. Here, the RECIP operator should be taken as a unit. It can float, just like a quantifier does, and it cannot be expressed (in the general case) in terms of duplex conditions – see (Dalrymple et al., 1998) for various meanings that a reciprocal could have, and (Peters and Westerståhl, 2006, section 10.4) for why some of those are not definable in terms of FOL quantification.

In this paper, I will show how choosing to use Situation Semantics as the underlying model-theoretic semantics for the above representations solves this issue because both variables turn out to be parameters. Since they are on an equal footing, they can be equated without posing the problem above. Along the way, I will show the details of the syntax-semantics interface that are necessary for computing the representations.

2 Calculating DRS Representations

2.1 Standard Ways

How are DRSs calculated from the syntactic analysis of a sentence? Bos et al. (1994) present λ-DRS, a proposal to extend the syntax-to-semantics mappings to handle DRSs. Here is a simple λ-DRS lexicon:

(6)

John	$\lambda P.[l \mid name(l) = \text{`}John\text{'}] \oplus P(l)$
likes	$\lambda y \lambda x.[e \mid like(e),\ subj(e, x),\ obj(e, y)]$
book	$\lambda x.[\ \mid book(x)]$
a	$\lambda P \lambda Q.[l \mid\ ] \oplus P(l) \oplus Q(l)$
every	$\lambda P \lambda Q.[\ \mid ([l \mid\ ] \oplus P(l)) \Rightarrow Q(l)]$
that	$\lambda Q \lambda P \lambda x.P(x) \oplus Q(x)$ (relative clause)

Semantic pieces are combined by using an extended functional composition:

(7) $\phi \odot \psi \ := \ \lambda \overrightarrow{\sigma}.\phi(\lambda v.(\psi(v))(\overrightarrow{\sigma}))$

This operator extends standard function composition $\lambda v.\phi(\psi(v))$ by accepting a n-place predicate ψ and binding only its first argument, while abstracting over the remaining $n - 1$ arguments in $\overrightarrow{\sigma}$.

Example derivation:

(8) John read a book.

- a $\odot$ book =
  $(\lambda P \lambda Q.([l \mid] \oplus P(l) \oplus Q(l)))(\lambda x.[\mid book(x)]) =$
  $\lambda Q.([l \mid] \oplus (\lambda x.[\mid book(x)])(l) \oplus Q(l)) =$
  $\lambda Q.([l \mid] \oplus [\mid book(l)] \oplus Q(l)) =$
  $\lambda Q.([l \mid book(l)] \oplus Q(l))$
- (a $\odot$ book) $\odot$ read =
  $(\lambda Q.([l \mid book(l)] \oplus Q(l)))(\lambda y \lambda x.[e \mid read(e),\ subj(e,x),\ obj(e,y)]) =$
  $\lambda x.(\lambda Q.([l \mid book(l)] \oplus Q(l)))(\lambda y.[e \mid read(e),\ subj(e,x),\ obj(e,y)]) =$
  $\lambda x.([l \mid book(l)] \oplus (\lambda y.[e \mid read(e),\ subj(e,x),\ obj(e,y)])(l)) =$
  $\lambda x.([l \mid book(l)] \oplus [e \mid read(e),\ subj(e,x),\ obj(e,l)]) =$
  $\lambda x.([l, e \mid book(l),\ read(e),\ subj(e,x),\ obj(e,l)])$
- John $\odot$ ((a $\odot$ book) $\odot$ read) =
  $(\lambda P.[l_2 \mid name(l_2) = \text{`John'}] \oplus P(l_2))(\lambda x.([l, e \mid book(l),\ read(e),\ subj(e,x),\ obj(e,l)])) =$
  $[l_2 \mid name(l_2) = \text{`John'}] \oplus (\lambda x.([l, e \mid book(l),\ read(e),\ subj(e,x),\ obj(e,l)]))(l_2) =$
  $[l_2 \mid name(l_2) = \text{`John'}] \oplus ([l, e \mid book(l),\ read(e),\ subj(e,l_2),\ obj(e,l)]) =$
  $[l, l_2, e \mid name(l_2) = \text{`John'},\ book(l),\ read(e),\ subj(e,l_2),\ obj(e,l)])$

This idea was later revised using Hole Semantics (Reyle, 1993; Blackburn and Bos, 2005) in order to deal with scope ambiguities.

2.2 Existing Suggestions for Glue-DRT

The above is essentially the same as a very basic and simple lambda-based semantic composition, except that λ-DRSs are used instead of simpler meaning terms. This simple scheme has shortcomings which I discussed in sections 3.1.4 and 7.6 of (Lev, 2007). Instead, we can use glue semantics (Dalrymple, 2001) for the composition because it is compatible with different meaning representation languages, including DRSs. This idea was first proposed in (van Genabith and Crouch, 1999; Kokkonidis, 2005), and here I present a slight variation on it. Below, I will assume familiarity with the basic glue semantics specification from chapters 3 and 4 of (Lev, 2007).

In the λ-DRT lexicon, instead of relying on standard lambda-composition, we can rely on the usual glue categories:

(9)

John	$\lambda P.[\hat{l} \mid name(\hat{l}) = \text{`John'}] \oplus P(\hat{l})\ :\ (l^e \to H^t) \to H^t$	
	where l is the label of my NP	
likes	$\lambda x \lambda y.[e \mid like(e),\ subj(e,x),\ obj(e,y)]\ :\ a^e \to b^e \to l^h$	
	where a is the label of my subject, b is the label of my object,	
	and l is the label of my clause.	
book	$\lambda x.[\mid book(x)]\ :\ {l_v}^e \to {l_r}^t$	
	where l is the label of my NP	
a	$\lambda P \lambda Q.[\hat{l} \mid] \oplus P(\hat{l}) \oplus Q(\hat{l})\ :\ ({l_v}^e \to {l_r}^t) \to (l^e \to k^t) \to k^t$	
	where l is the label of my NP and k is the label of the clause I scope at	
every	$\lambda P \lambda Q.[([\hat{l} \mid] \oplus P(\hat{l})) \Rightarrow Q(\hat{l})]\ :\ ditto$	
that	$\lambda Q \lambda P \lambda x.P(x) \oplus Q(x)\ :\ (a^e \to b^t) \to ({l_v}^e \to {l_r}^t) \to {l_v}^e \to {l_r}^t$	(relative clause)

The hat $\hat{}$ is a function that takes a basic glue label and returns a discourse variable. It allows us to link glue categories and discourse variables. This is useful because we have constraints on possible anaphoric links which are expressed in

terms of positions in the syntactic C- and F-structures. Those constraints induce constraints on anaphoric links between glue categories, and those in turn, through the hat function, induce constraints on equations between discourse variables.

Thus, if a NP with glue category f precedes a NP with glue category g in the sentence, we can add the constraint *not-antecedent*(g, f) (g is not a possible antecedent of f if f is an anaphoric expression), and this constraint will later prevent us from equating $\hat{f}$ with $\hat{g}$. In addition, syntactic binding theory gives us constraints such as: a reflexive pronoun must be anaphoric to a NP in the same minimal clause, while a non-reflexive pronoun cannot co-refer with a NP in the same minimal clause. These constraints are similarly expressed in terms of glue categories, and affect the possible equations between the corresponding discourse variables.

2.3 My Version

While this solution works, it is very tedious to require writing DRS structures and $\oplus$ operators for every entry – many entries have nothing to do with anaphora (see e.g. *book* and *that*). Also notice the treatment of proper names as scoping quantifiers rather than non-scoping terms (see the discussion about this in (Lev, 2007, section 3.1.4)). It would also be nice if we did not have to revise all the glue entries that have already been developed before anaphora was considered. What we really want is to keep what we have done, and change only the parts that have something to do with anaphora. For example, we do not want to change our treatment of nouns and relative clauses. Even for verbs, if we decide for now not to handle expressions that are anaphoric to event variables, we can keep our old specifications for verbs.

My aim is that at the end of the glue composition phase, we will end up with a representation that looks something like this:

(10) Every man that saw Mary liked her.

$$\begin{aligned}&every(\lambda x.man(x) \wedge \exists e_1.see(e_1) \wedge subj(e_1, x) \wedge \\ &\qquad obj(e_1, entity(l_1, \{name(l_1) = \text{`}Mary\text{'}, female(l_1), individual(l_1)\})), \\ &\quad \lambda x.\exists e_2.like(e_2) \wedge subj(e_2, x) \wedge obj(e_2, dref(l_2, \{individual(l_2), female(l_2)\})))\end{aligned}$$

Notice that overall the representation looks like a simple, non-DRS representation, and only certain pieces mention discourse variables. The operators *entity* and $dref$ could be seen as shorthands as follows:

(11) $entity(v, S) \equiv [\langle v, S\rangle \mid v]$
$dref(v, S) \equiv [\langle v, (\{v = ?\} \cup S)\rangle \mid v]$

In these definitions, I use a slightly different version of DRSs. Instead of a pair consisting of a set of discourse variables and a set of conditions, I have a pair whose second element is just one meaning expression and whose first element is a set of *conditioned* discourse variables, where a conditioned discourse variables is a pair consisting of a discourse variable and a set of associated conditions. In (10), the discourse variable l_1 has three conditions associated with it, including $name(l_1) =$'*Mary*'. I use this form to make it easier to locate the constraints that are relevant for each discourse variable.

In a certain sense, I am already getting closer to Situation Semantics here. The v in these definitions can be seen as a parameter that has restrictions on it. In the case of $dref$, there is a restriction $v =?$, whose resolution (equating it to another parameter) is determined by the discourse context.

The *entity* and $dref$ expressions act locally as a v in the expression where they appear. The discourse variable introduced by an *entity* expression should be accessible everywhere, but the *entity* expression is written *in situ* rather than in a global place to make it easier to write the glue semantics specification (the information will be moved up by the anaphora resolution algorithm below). The information in a $dref$ expression should be used by the anaphora resolution module when the pronoun is resolved and the $dref$ expression is replaced by another variable.

In light of this, we change the glue rule for proper names from (Lev, 2007) to:

(12) proper name n of gender g:
$entity(\hat{l}, \{name(\hat{l}) = n,\ gender(\hat{l}) = g\})\ :\ l^e$
where l is the label of my NP

We also add for pronouns:

(13) pronoun with gender g and number n (sg or pl):
$dref(\hat{l}, \{gender(\hat{l}) = g,\ number(\hat{l}) = n\})$

And that's it. There is no need to make any more changes to the glue specification. All the rest of the work will be done by the anaphora resolution module, as described below. This includes treating quantifier expressions as a shorthand for the appropriate DRS structures. Thus, we get a modular solution: The specification of composition is simple, supplying just enough information needed to handle anaphora. Anaphora resolution using DRSs is done in a separate module, and the writer of the glue specification does not need to worry about using DRSs explicitly. The justification for this approach is that it produces the same results that would be obtained if we followed the more explicit glue-DRT approach above.

3 Resolving the Representations

Since the glue specification was left mostly unchanged, I need to define here how to expand some of the expressions, such as the quantifiers, and how to deal with *entity* and $dref$ expressions.

3.1 Preparing

3.1.1 Entity Raising

The conditions inside an *entity* expression should be moved to the topmost DRS because there should be no DRS-accessibility restrictions on proper names serving as antecedents of anaphoric expressions.[2] Therefore, every $entity(l, S)$ in φ is replaced

[2]This is not the most general treatment because in some sentences in NL, proper names may not be so accessible. For example: "The John from our street met the John that was elected as Mayor." However, this treatment will do for now.

with l to obtain φ'. Additionally, for every such entity, the pair $\langle l, S\rangle$ is added to a collection C. Then, a top DRS is created:

(14) $[C \mid \varphi']$

For example,

(15) John saw Mary.
$\exists e.see(e) \wedge subj(e, entity(l_1, \{name(l_1) = \text{'John'}, gender(l_1) = male, num(l_1) = sg\})) \wedge$
$obj(e, entity(l_2, \{name(l_2) = \text{'Mary'}, gender(l_2) = female, num(l_2) = sg\}))$
$\Rightarrow$
$[\langle l_1, \{name(l_1) = \text{'John'}, gender(l_1) = male, num(l_1) = sg\}\rangle,$
$\langle l_2, \{name(l_2) = \text{'Mary'}, gender(l_2) = female, num(l_2) = sg\}\rangle \mid$
$\exists e.see(e) \wedge subj(e, l_1) \wedge obj(e, l_2)]$

3.1.2 Quantifier Expansion

(16) $tr(a(P, Q)) = [\langle l, tr(P(l))\rangle \mid tr(Q(l))]$
$tr(every(P, Q)) = [\ |\ (\ [\langle l, tr(P(l))\rangle \mid\ ] \Rightarrow [\ |\ tr(Q(l))]\ )\ ]$
$tr(no(P, Q)) = [\ |\ \neg tr(a(P, Q))]$

3.1.3 Merge Embedded DRS

If the body of a DRS is itself a DRS, the two can be merged:

(17) $tr([V_1 \mid [V_2 \mid B]]) = tr([V_1 \cup V_2 \mid B])$

3.2 Resolution of *dref*

After all the previous transformations are done, we obtain a normal DRS, and we are ready to resolve anaphora in it. We do this by recursively traversing top-down the DRS for the entire text. At each step, the possible antecedents of a pronoun are those that are possible according to the DRS accessibility relations. These are remembered in a second argument to the *resolve* predicate as follows:

(18) $\textit{start-resolve}(\varphi) = resolve(\varphi, \{\})$
$resolve([V|D], S) = resolve(D, V \cup S)$
$resolve(([V|B] \Rightarrow D), S) = resolve([V|B], S) \Rightarrow resolve(D, V \cup S)$

Notice that when we enter the right-hand-side of a DRS implication, we add the discourse variables of the implication's antecedent to the list of possible antecedents. This is in accordance with the definition of DRS accessibility and allows us to resolve donkey anaphora correctly.

Furthermore, the algorithm takes into account the gender constraints, as they are written in the conditions associated with the discourse variables. We also take into account the *not-antecedent* and *not-eq* constraints from section 2.2. For example, $\textit{not-ant}(g, h)$ prevents the algorithm from resolving a $dref(\hat{h}, \ldots)$ to $\hat{g}$.

There may be more than one possible resolution to a $dref$ expression. In that case, we can find and return all possibilities. If a $dref$ cannot be resolved to any

discourse variable, then the particular interpretation that is processed is incorrect. For example, consider:

(19) a. Every man likes a woman. She is happy.
b. $every(man, \lambda x.a(woman, \lambda y.like(x,y))) \oplus happy(dref(l, \{female(l)\}))$
c. $a(woman, \lambda y.every(man, \lambda x.like(x,y))) \oplus happy(dref(l, \{female(l)\}))$

The first sentence in (19)a has a scope ambiguity, so the two possible resulting representations are (19)b,c. These are converted to the DRS representations:

(20) a. $[\ |\ ([\langle l_1, man(l_1)\rangle\ |\ ] \Rightarrow [\langle l_2, woman(l_2)\rangle\ |\ like(l_1,l_2)]),\ happy(dref(l, \{female(l)\}))]$
b. $[\langle l_2, woman(l_2)\rangle\ |\ ([\langle l_1, man(l_1)\rangle\ |\ ] \Rightarrow [|\ like(l_1,l_2)]),\ happy(dref(l, \{female(l)\}))]$

In the first representation, l_2 is not accessible to l, and as there is no other possibility, $dref$ cannot be resolved. In the second representation, l_2 appears in the top DRS and so is accessible to l, so that representation can be resolved to:

(21) $[\langle l_2, \{woman(l_2), female(l)\}\rangle\ |\ ([\langle l_1, man(l_1)\rangle\ |\ ] \Rightarrow [|\ like(l_1,l_2)]),\ happy(l_2)]$

4 Plurals and Anaphora

4.1 Plurals and Ambiguity

Following (Link, 1998), I will assume that the domain includes pluralities, which are constructed using the summation operator $\oplus$. Thus, the initial meaning representation of (22)a is (22)b.

(22) a. John, Bill, and Mary sneezed.
b. $(\lambda x.sneeze(x))^{\square}(john \oplus bill \oplus mary)$

The $^{\square}$ here signifies that the formula is underspecified regarding whether the predicate is applied on the plurality collectively or distributively. In the first case, the $^{\square}$ can just be dropped. In the second case, $P^{\square}(a)$ is resolved to $\forall x.x \in a \rightarrow P(x)$. The $^{\square}$ is always used with a plural NP. In the case of the verb *sneeze*, it is always disambiguated to a distributive application. With a collective predicate such as *meet*, it is always disambiguated to a collective application. With an ambiguous predicate such as *weigh 200 lbs*, both options are possible.

4.2 Conjoined Names

The F-structure of a conjunction of names is, for example:

(23) John, Bill, and Mary sneezed.

$$\begin{bmatrix} \text{PRED} & \text{`SNEEZE}\langle\text{SUBJ}\rangle\text{'} \\ \text{SUBJ} & \begin{bmatrix} \text{COORD} & \text{AND} \\ \text{COORD-LEVEL} & \text{NP} \\ \left\{ \begin{bmatrix} \text{PRED} & \text{`JOHN'} \end{bmatrix}, \begin{bmatrix} \text{PRED} & \text{`BILL'} \end{bmatrix}, \begin{bmatrix} \text{PRED} & \text{`MARY'} \end{bmatrix} \right\} \end{bmatrix} \end{bmatrix}$$

The glue specification is:

(24) A coordination NP whose label is l
where (l COORD) = AND and (l COORD-LEVEL) = NP:
$\lambda x.\oplus x \ : \ conj(l)^e \rightarrow l^e$
For the rightmost conjunct, whose label is k:
$\lambda x.\{x\} \ : \ k^e \rightarrow conj(l)^e$
For any other conjunct, whose label is k:
$\lambda x \lambda y.(\{x\} \cup y) \ : \ k^e \rightarrow conj(l)^e \rightarrow conj(l)^e$

Note that $\{x\} \cup y$ is intended to be *evaluated* during the construction of the expression for the NP in the glue derivation. Thus, at the end of the glue derivation, we will get (25)b rather than (25)a.

(25) "John, Bill, and Mary"
a. $(\{john\} \cup (\{bill\} \cup \{mary\}))$
b. $\oplus\{john, bill, mary\}$ (shorthand for: $john \oplus bill \oplus mary$)

This was a simplification because just as with a single name in (12), we want to create an *entity* expression that could be the antecedent of other anaphoric expressions. Furthermore, that *entity* expression acts as a kind of quantifier, represented by the $^{\square}$, because in the distributive case, we want to quantify over the members of the plurality:

(26) $(\lambda x.sneeze(x))^{\square}(T)$
where:
$T = entity(\hat{l}_1, \{\hat{l}_1 = \oplus(\{entity(\hat{l}_2, \{name(\hat{l}_2) = \text{`John'}\}),$
$entity(\hat{l}_3, \{name(\hat{l}_3) = \text{`Mary'}\}),$
$entity(\hat{l}_4, \{name(\hat{l}_4) = \text{`Bill'}\})\})\})$

To get this, we revise the first line of (24) to:

(27) $\lambda x \lambda P.\ P^{\square}(entity(\hat{l}, \{\hat{l} = \oplus(x)\})) \ : \ conj(l)^e \rightarrow (l^e \rightarrow H^t) \rightarrow H^t$

5 The Problem with Composite Predicates

Consider again sentence (1):

(28) John and Mary think they will win.

Using the above machinery, we get:

(29) think they will win =
$\lambda x.think(x, win(dref(l_4, num(l_4) = pl)))$
John and Mary =
$\lambda P.\ P^{\square}(entity(l_1, l_1 = \oplus(\{entity(l_2, l_2 = john), entity(l_3, l_3 = mary)\})))$
John and Mary think they will win. =
$(\lambda x.think(x, win(dref(l_4, num(l_4) = pl))))^{\square}(entity(l_1, l_1 = \oplus(\{entity(l_2, l_2 = john),$
$entity(l_3, l_3 = mary)\})))$

With entity raising, we get:

(30) $[\langle l_1, \{l_1 = \oplus(l_2, l_3)\}\rangle, \langle l_2, \{l_2 = john\}\rangle, \langle l_3, \{l_3 = mary\}\rangle \mid (\lambda x.think(x, win(dref(l_4, num(l_4) = pl))))^{\Box}(l_1)]$

5.1 Resolution of *dref* to an Abstracted Variable

Now you can see that we must allow l_4 to be resolved to x, to get:

(31) $[\langle l_1, \{l_1 = \oplus(l_2, l_3)\}\rangle, \langle l_2, \{l_2 = john\}\rangle, \langle l_3, \{l_3 = mary\}\rangle \mid (\lambda x.think(x, win(x)))^{\Box}(l_1)]$

And once we resolve $^{\Box}$ to the distributive reading, we get:

(32) $[\langle l_1, \{l_1 = \oplus(l_2, l_3)\}\rangle, \langle l_2, \{l_2 = john\}\rangle, \langle l_3, \{l_3 = mary\}\rangle \mid \forall y \in l_1.think(y, win(y))]$

So we need to revise the algorithm in section 3.2 to allow a *dref* to be resolved to a lambda variable that appears somewhere above it.

I already explained in section 1 that this resolution poses a problem because according to the MTS definitions, *dref* should be resolved to a discourse variable, which denotes a dynamic discourse variable in the domain, and not to a normal variable like x.

5.2 Situation Semantics Solution

A possible way to make the equating of the two variables "kosher" is to make both of them parameters in the sense of Situation Semantics (Barwise and Perry, 1983; Devlin, 1991, 2006; Gawron and Peters, 1990). I present here a simplified version of Situation Semantics (only what is really needed for the solution), and the reader is invited to consult those sources for further details. In this simplified version, the meaning of a sentence is taken to consist of a *structured object* called *infon*. We can then further check whether a particular situation in a particular context *supports* this infon (the claim that the sentence is making is true in the situation) or not (the claim is false in the situation) or neither (the situation is silent on that matter). Furthermore, the structured object may contain certain objects called *parameters*, in which case it is a parametric object. Practitioners of Situation Semantics go to great pains to emphasize that a parameter is not a variable in the formal language but an element of the domain. Therefore, a whole new mathematical notation for talking about the structured objects is used.

An important difference between the lambda calculus and Situation Semantics is this: In the former, (33)a and (33)b are two different expressions in the formal language which nonetheless denote the same mathematical object. In contrast, (33)c and (33)d denote two different mathematical objects (in this case, infons) in Situation Semantics.

(33) assume t can be freely substituted for x in φ
- a. $(\lambda x.\varphi)(r)$ (e.g.: $(\lambda x.sleep(x))(r)$)
- b. $\varphi[r/x]$ (e.g.: $sleep(r)$)

c. $\langle\langle [p \mid \varphi],\ r \rangle\rangle$ (e.g.: $\langle\langle [p \mid \langle\langle sleep, p \rangle\rangle],\ r \rangle\rangle$)
d. $\varphi[r/p]$ (e.g.: $\langle\langle sleep, r \rangle\rangle$)

If we wish, we could say that (under normal conditions), if (33)c and (33)d are infons, then one is supported by a situation iff the other is. However, we do not have to make this move.

Using the notation of Situation Semantics,[3] we could say that the meaning of the sentence (28) should be represented as (34) rather than (29).

(34) $\langle\langle\ [p \mid \langle\langle think,\ p,\ \langle\langle win, q_{they} \rangle\rangle\rangle\rangle]\ ,\ r_\sigma\ \rangle\rangle$
where σ is $\langle\langle =,\ r,\ s_{John} \oplus t_{Mary} \rangle\rangle$

The terms p, q, r, s, and t are all parameters, not variables.[4] A parameter may have restrictions on the possible values it could refer to. Thus, s_{John} has the restriction that the parameter may refer only to a human entity whose name is "John." So now the denotation of the semantic representations we assign to sentences are not functions and predicates but rather structured mathematical objects from the Situation Semantics ontology.

How does all this help us? First, we do not necessarily need to use the $\Box$ mechanism. This is possible if we do not say that (33)c is supported by a situation iff (33)d is, and instead say that this holds only in some contexts. In other contexts, (33)c is supported in a situation iff the following infon is supported there:

(35) $\langle\langle forall,\ u,\ \langle\langle imply,\ \langle\langle \in, u, r \rangle\rangle,\ \varphi[u/p] \rangle\rangle\rangle\rangle$

Second, p and q in (34) are both parameters of exactly the same status (and therefore can be legitimately equated), whereas x and $dref$ had a different status in (29). Here is how it works. Suppose that in a particular context C_1, we are given further information that the parameter q has a restriction $\langle\langle =,\ q,\ a \rangle\rangle$ for some group object a that was mentioned previously in the context. Then in context C_1, the sentence (28) means that both John and Mary think that the group a will win. In another context C_2, we are given the information that the parameter q has a restriction $\langle\langle =,\ q,\ p \rangle\rangle$. So in C_2, the proposition's meaning is:

(36) $\langle\langle\ [p \mid \langle\langle think,\ p,\ \langle\langle win, q_{\langle\langle =, q, p \rangle\rangle} \rangle\rangle\rangle\rangle]\ ,\ r_\sigma\ \rangle\rangle$
where σ is $\langle\langle =,\ r,\ s_{John} \oplus t_{Mary} \rangle\rangle$

Because p is abstracted over, (36) denotes the same object as (37) under the rules of Situation Semantics.

(37) $\langle\langle\ [p \mid \langle\langle think,\ p,\ \langle\langle win, p \rangle\rangle\rangle\rangle]\ ,\ r_\sigma\ \rangle\rangle$
where σ is $\langle\langle =,\ r,\ s_{John} \oplus t_{Mary} \rangle\rangle$

This is what we wanted to get. If we further assume that C_2 is one of those contexts where (33)c is supported in a situation iff (35) is supported, then we get that in C_2, the following is supported:

[3] I am using here a simplified version of the notation in (Gawron and Peters, 1990).

[4] If we considered (34) to be an expression in something like a sorted first-order language, we would say that each of p, q, r, s, and t is an object constant, whose denotation is an element (a member of the domain) of type "parameter".

(38) $\langle\!\langle forall,\ u,\ \langle\!\langle imply,\ \langle\!\langle \in, u, r_\sigma \rangle\!\rangle,\ \langle\!\langle think,\ u,\ \langle\!\langle win, u \rangle\!\rangle \rangle\!\rangle \rangle\!\rangle \rangle\!\rangle \rangle\!\rangle$
where σ is $\langle\!\langle =,\ r,\ s_{John} \oplus t_{Mary} \rangle\!\rangle$

Let us recap. We started with a representation (29). We made some transformations to obtain (30). These transformations were justified by the intended model-theoretic semantics (MTS). However, the transformation from that to (31) was not justified by the MTS because we equated a language variable with a discourse variable. The solution was to provide a more complex MTS that uses Situation Semantics objects. In this MTS, equating p and q was justified since they were both of the same kind (parameters).

Is this move illuminating? Some would say yes. Furthermore, Situation Semantics has several other independent motivations, and is needed when analyzing more complex intensional and context-dependant linguistic constructions, so if one is using that framework, one might as well use the solution proposed here.

When moving to Situation Semantics, we essentially decide to rely on some distinctions between representations which previously we considered to be equivalent. If one takes this process to its extreme, we eventually decide that no two representations are inherently equivalent. This kind of move might in fact be unavoidable given the nature of natural language. Two distinct expressions that might at first seem equivalent in all respects may later turn out to have some subtle differences in meaning, which surface in different contexts, and so the two expressions should be given similar but distinguishable meanings (and representations).

5.3 Resolution of a Singular $dref$, Revisited

Now that we know that a plural $dref$ must be allowed to refer to an abstracted variable, should we allow this for a singular $dref$ as well? In (39), this would give us (39)c rather than (39)b.

(39) a. John thinks that he is smart.
b. $think(entity(\hat{l}_1, \{name(\hat{l}_1) = \text{`}John\text{'}\}), smart(dref(\hat{l}_2, sg, male))))$
c. $(\lambda x.think(x, smart(dref(\hat{l}_2, sg, male))))^{app}(entity(\hat{l}_1, \{name(\hat{l}_1) = \text{`}John\text{'}\}))$

This requires complicating the entries of NPs. For example, we need (40)b instead of (40)a.

(40) proper name n
a. $entity(\hat{l}, \{name(\hat{l}) = n\})\ :\ l^e$ where l is my NP's label
b. $\lambda P.P(entity(\hat{l}, \{name(\hat{l}) = n\}))\ :\ (l^e \rightarrow H^t) \rightarrow H^t$
where l is my NP's label and H^t is the topmost clause

According to some theories of ellipsis, e.g. (Gawron and Peters, 1990), both the bound reading (39)c and the direct reading (39)b are in fact needed, as they provide the "sloppy" and "strict" readings of sentences such as:

(41) John thinks he is smart. Bill does too.

If (39)b is used, then the predicate $\lambda s.think(x, smart(john))$ is applied on Bill in the second sentence to get the strict reading. If (39)c is used, then the predicate $\lambda s.think(x, smart(x))$ is first created in the first sentence and so it is available in the context to be applied on Bill in the second sentence, giving the sloppy reading.

5.4 The Solution Works for Reciprocals Too

Consider the sentence:

(42) John and Mary think they like each other.

I gave an analysis of it in (Lev, 2006). The entry for *they* which I used there was in the line of using glue semantics derivations for doing anaphora resolution, as described in section 8.2 of (Lev, 2007). However, that section explains in detail why using glue semantics derivations for anaphora resolution is not a good idea, and instead, the Glue-DRT-based solution is better. Therefore, I will now recreate the analysis of (42) in those terms, and this solves the problem that I mentioned in footnote 9 of (Lev, 2006).

Instead of (22) in (Lev, 2006), we now have:

(43) [[John and Mary]$_2$ think [[they]$_4$ like [each other]$_5$]$_3$]$_1$

$\lambda P.P^{\Box}(entity(l_1, l_1 = \oplus(\{entity(l_2, l_2 = john),$ $entity(l_3, l_3 = mary)\})))$	$: (\mathbf{2}^e \rightarrow H^t) \rightarrow H^t$
$think$	$: \mathbf{2}^e \rightarrow \mathbf{3}^t \rightarrow \mathbf{1}^t$
$dref(\hat{\mathbf{4}}^e, \{num(\hat{\mathbf{4}}^e) = pl\})$	$: \mathbf{4}^e$
$like$	$: \mathbf{4}^e \rightarrow \mathbf{5}^e \rightarrow \mathbf{3}^t$
$\lambda R \lambda z.\textsc{recip}(z, R)$	$: (\mathbf{5}_{ant}{}^e \rightarrow \mathbf{5}^e \rightarrow l^t) \rightarrow \mathbf{5}_{ant}{}^e \rightarrow l^t$

The first reading of (43) in which RECIP takes narrow scope can only be obtained if $l = \mathbf{3}$ and $\mathbf{5}_{ant} = \mathbf{4}$. We then get:

(44) $\lambda R \lambda z.\textsc{recip}(z, R) : (\mathbf{4}^e \rightarrow \mathbf{5}^e \rightarrow \mathbf{3}^t) \rightarrow \mathbf{4}^e \rightarrow \mathbf{3}^t$
$+like \Rightarrow \lambda z.\textsc{recip}(z, like) : \mathbf{4}^e \rightarrow \mathbf{3}^t$
$+they \Rightarrow \textsc{recip}(dref(\hat{\mathbf{4}}^e, ..), like) : \mathbf{3}^t$
$+think \Rightarrow \lambda u.think(u, \textsc{recip}(dref(\hat{\mathbf{4}}^e, ..), like)) : \mathbf{2}^e \rightarrow \mathbf{1}^t$
$+jm \Rightarrow (\lambda u.think(u, \textsc{recip}(dref(\hat{\mathbf{4}}^e, ..), like)))^{\Box}(entity(l_1, ..)) : \mathbf{1}^t$

With entity raising, we get:

(45) $[\langle l_1, \{l_1 = \oplus(l_2, l_3)\}\rangle, \langle l_2, \{l_2 = john\}\rangle, \langle l_3, \{l_3 = mary\}\rangle \mid$
$(\lambda u.think(u, \textsc{recip}(dref(\hat{\mathbf{4}}^e, ..), like)))^{\Box}(l_1)]$

Now, $^{\Box}$ should be resolved distributively (because of world knowledge, we know that *think* can hold only on individuals and not on collectives, unless we are in a science fiction situation). Also, because $dref(\hat{\mathbf{4}}^e, ..)$ is the first argument of RECIP, it must not be resolved to an individual, so its antecedent cannot be u and must be l_1. This gives us:

(46) $[\langle l_1, \{l_1 = \oplus(l_2, l_3)\}\rangle, \langle l_2, \{l_2 = john\}\rangle, \langle l_3, \{l_3 = mary\}\rangle \mid$
$\forall y \in l_1.think(y, \textsc{recip}(l_1, like))]$
= John thinks 'John and Mary like each other', and Mary thinks the same.

The second reading of (43) in which RECIP takes wide scope can only be obtained if $l = \mathbf{1}$ and $\mathbf{5}_{ant} = \mathbf{2}$.

(47) $\lambda R\lambda z.\text{RECIP}(z,R) : (\mathbf{2}^e \to \mathbf{5}^e \to \mathbf{1}^t) \to \mathbf{2}^e \to \mathbf{1}^t$
$they + like \Rightarrow \lambda y.like(dref(\hat{\mathbf{4}}^e,..),y) : \mathbf{5}^e \to \mathbf{3}^t$
$+think \Rightarrow \lambda x\lambda y.think(x,like(dref(\hat{\mathbf{4}}^e,..),y)) : \mathbf{2}^e \to \mathbf{5}^e \to \mathbf{1}^t$
$+recip \Rightarrow \lambda z.\text{RECIP}(z,\lambda x\lambda y.think(x,like(dref(\hat{\mathbf{4}}^e,..),y))) : \mathbf{2}^e \to \mathbf{1}^t$
$+jm \Rightarrow (\lambda z.\text{RECIP}(z,\lambda x\lambda y.think(x,like(dref(\hat{\mathbf{4}}^e,..),y))))^{\square}(entity(l_1,..)) : \mathbf{1}^t$

With entity raising, we get:

(48) $[\langle l_1,\{l_1 = \oplus(l_2,l_3)\}\rangle, \langle l_2,\{l_2 = john\}\rangle, \langle l_3,\{l_3 = mary\}\rangle \mid$
$(\lambda z.\text{RECIP}(z,\lambda x\lambda y.think(x,like(dref(\hat{\mathbf{4}}^e,..),y))))^{\square}(l_1)]$

Now, $^{\square}$ must be resolved collectively because z is the first argument of RECIP. If $dref(\hat{\mathbf{4}}^e,..)$ is resolved to l_1 then we get the (less likely) reading where John thinks that John and Mary like Mary and Mary thinks that John and Mary like John. This time, however, $dref(\hat{\mathbf{4}}^e,..)$ can also be resolved to x, just like $dref(l_4)$ was resolved to x in (30)-(31). Since x is 'closer' than l_1 to $dref(\hat{\mathbf{4}}^e)$, this resolution is preferred, giving us:

(49) $[\langle l_1,\{l_1 = \oplus(l_2,l_3)\}\rangle, \langle l_2,\{l_2 = john\}\rangle, \langle l_3,\{l_3 = mary\}\rangle \mid$
$\text{RECIP}(l_1,\lambda x\lambda y.think(x,like(x,y)))]$
= John thinks that he likes Mary and Mary thinks she likes John.

What prevents $dref(\hat{\mathbf{4}}^e)$ from being resolved to y? In the sentence, $dref$ originates from the NP *they*, which linearly precedes the NP *each other* from which y originates. Therefore we have a constraint that *they* cannot be anaphoric to the later NP. This constraint in terms of syntactic positions induces a constrain in terms of glue categories. All we then need is to remember the glue category from which each variable came. Thus, y came from the NP *each other* and so cannot serve as the antecedent of *they*, whereas x came from the NP *John and Mary* and can serve as the antecedent.

What about the two remaining options for the values of l and $\mathbf{5}_{ant}$? If $l = \mathbf{3}$ and $\mathbf{5}_{ant} = \mathbf{2}$, then the premises cannot combine in any glue derivation.[5] However, if $l = \mathbf{1}$ and $\mathbf{5}_{ant} = \mathbf{4}$, we actually get a third reading:[6]

(50) $\lambda R\lambda z.\text{RECIP}(z,R) : (\mathbf{4}^e \to \mathbf{5}^e \to \mathbf{1}^t) \to \mathbf{4}^e \to \mathbf{1}^t$
$think + like \Rightarrow \lambda z\lambda x\lambda y.think(z,like(x,y)) : \mathbf{2}^e \to \mathbf{4}^e \to \mathbf{5}^e \to \mathbf{1}^t$
$+recip \Rightarrow \lambda z\lambda u.\text{RECIP}(u,\lambda x\lambda y.think(z,like(x,y))) : \mathbf{2}^e \to \mathbf{4}^e \to \mathbf{1}^t$
$+they \Rightarrow \lambda z.\text{RECIP}(dref(\hat{\mathbf{4}}^e,..),\lambda x\lambda y.think(z,like(x,y))) : \mathbf{2}^e \to \mathbf{1}^t$
$+jm \Rightarrow (\lambda z.\text{RECIP}(dref(\hat{\mathbf{4}}^e,..),\lambda x\lambda y.think(z,like(x,y))))^{\square}(entity(l_1,..)) : \mathbf{1}^t$

With entity raising, we get:

(51) $[\langle l_1,\{l_1 = \oplus(l_2,l_3)\}\rangle, \langle l_2,\{l_2 = john\}\rangle, \langle l_3,\{l_3 = mary\}\rangle \mid$
$(\lambda z.\text{RECIP}(dref(\hat{\mathbf{4}}^e,..),\lambda x\lambda y.think(z,like(x,y))))^{\square}(l_1)]$

[5] To see this, note that $\mathbf{4}^e$ would have to combine with $\mathbf{4}^e \to \mathbf{5}^e \to \mathbf{3}^e$ to give $\mathbf{5}^e \to \mathbf{3}^e$. But this cannot fit as the first argument of RECIP : $(\mathbf{2}^e \to \mathbf{5}^e \to \mathbf{3}^t) \to \mathbf{2}^e \to \mathbf{3}^t$. It will not help to try to first combine $\mathbf{5}^e \to \mathbf{3}^e$ with $\mathbf{2}^e \to \mathbf{3}^e \to \mathbf{1}^t$.

[6] I missed this one in (Lev, 2006) but it's possible there too.

Since z is the first argument of *think*, it should be individual, and so $\square$ should be resolved distributively. In contrast, $dref(\hat{\mathbf{4}}^e)$ is the first argument of RECIP and should refer to a collection, so it cannot be resolved to z but only to l_1. Thus we get:

(52) $[\langle l_1, \{l_1 = \oplus(l_2, l_3)\}\rangle, \langle l_2, \{l_2 = john\}\rangle, \langle l_3, \{l_3 = mary\}\rangle \mid$
$\forall z \in l_1.\text{RECIP}(l_1, \lambda x \lambda y.think(z, like(x, y)))]$
= John thinks that John likes Mary and John thinks that Mary likes John and Mary thinks that John likes Mary and Mary thinks that Mary likes John.

This result is obviously different from (49), but it is also not the same as (46) because although the propositions $\text{RECIP}(a \oplus b, R)$ and $R(a, b) \wedge R(b, a)$ are logically equivalent, they are not the same proposition and not the same thought (a person thinking a proposition may not also think all its logically equivalent propositions). Is this third reading available for the sentence? It is hard to say. If we want to block it, we can require in the glue specification of a reciprocal that the clause over which the reciprocal scopes is the minimal clause containing the reciprocal's antecedent:

(53) $\lambda z \lambda R.\text{RECIP}(z, R) : a^e \to (a^e \to l^e \to k^t) \to k^t$
where l is my label, k is the label of the clause that I scope over,
and a is my antecedent's label;
where k = the label of the minimal clause containing a

This is a natural requirement – see Problem 3 in section 8.2.3 of (Lev, 2007) for a discussion of why this restriction is required also in the bound-anaphora analysis of pronouns.

(54)
$$\boxed{a}\begin{bmatrix}\text{PRED} & \text{SEE}\langle \boxed{b}, \boxed{g}\rangle \\ \text{SUBJ} & \boxed{b}\begin{bmatrix}\text{PRED} & \text{BILL} \\ \text{NTYPE} & \text{NAME}\end{bmatrix} \\ \text{OBJ} & \boxed{g}\begin{bmatrix}\text{SPEC} & \text{THE} \\ \text{PRED} & \text{GIRL} \\ \text{ADJUNCT} & \left\{\langle \text{A2: } \boxed{w}\rangle\right\}\end{bmatrix} \\ \text{ADJUNCT} & \left\{\langle \text{A1: } \boxed{w}\begin{bmatrix}\text{PRED} & \text{WITH}\langle \boxed{h}\rangle \\ \text{OBJ} & \boxed{h}\begin{bmatrix}\text{SPEC} & \text{THE} \\ \text{PRED} & \text{TELESCOPE}\end{bmatrix}\end{bmatrix}\rangle\right\}\end{bmatrix}$$

(55)
$$\boxed{a}\begin{bmatrix}\text{PRED} & \text{SEE}\langle \boxed{g}, \boxed{h}\rangle \\ \text{SUBJ} & \boxed{g}\begin{bmatrix}\text{SPEC} & \text{EVERY} \\ \text{PRED} & \text{MAN}\end{bmatrix} \\ \text{OBJ} & \boxed{h}\begin{bmatrix}\text{SPEC} & \text{SOME} \\ \text{PRED} & \text{WOMAN}\end{bmatrix}\end{bmatrix}$$

(56) $$\frac{\psi : A : S_1 \quad \delta : A_L \to B : S_2}{\delta(\lambda v_{i_1}, \ldots, \lambda v_{i_n}.\psi) : B : S_1 \cup S_2}$$ provided $S_1 \cap S_2 = \emptyset$ and $L \subseteq S_1$ and $L = [i_1, \ldots, i_n]$

(57) $$\frac{C_1 : \psi : A : PS_1 \qquad C_2 : \delta : A_L \rightarrow B : PS_2}{C_1 \wedge C_2 : \delta(\lambda v_{i_1}, \ldots, \lambda v_{i_n}.\psi) : B : PS}$$
provided $C_1 \wedge C_2 \neq 0$ and $complement(PS_1, PS_2)$ and $L \subseteq PS_1$ and $L = [i_1, \ldots, i_n]$ and $PS = union(C_1 \wedge C_2, PS_1, PS_2)$

(58) $$\frac{A : S_1 \qquad A \rightarrow B : S_2}{B : S}$$
provided $S_1 \cap S_2 = \emptyset$ and $S = S_1 \cup S_2$

(59) $$\frac{C_1 : A : PS_1 \qquad C_2 : A \rightarrow B : PS_2}{C_1 \wedge C_2 : B : PS}$$
provided $C_1 \wedge C_2 \neq 0$ and $complement(PS_1, PS_2)$ and $PS = union(C_1 \wedge C_2, PS_1, PS_2)$

References

Barwise, Jon, and John Perry. 1983. *Situations and attitudes*. Bradford Books – MIT Press.

Blackburn, Patrick, and Johan Bos. 2005. *Working with discourse representation theory: An advanced course in computational semantics*. http://www.blackburnbos.org/.

Bos, Johan, E. Mastenbroek, S. McGlashan, S. Millies, and M. Pinkal. 1994. A compositional DRS-based formalism for NLP applications. In *Proc. of the International Workshop on Computational Semantics*.

Carpenter, Bob. 1998. *Type-logical semantics*. MIT Press.

Dalrymple, Mary. 2001. *Lexical Functional Grammar*, volume 34 of *Syntax and Semantics Series*. Academic Press.

Dalrymple, Mary, Makoto Kanazawa, Yookyung Kim, Sam Mchombo, and Stanley Peters. 1998. Reciprocal expressions and the concept of reciprocity. *Linguistics and Philosophy* 21:159–210.

Devlin, Keith. 1991. *Logic and information*. Cambridge University Press.

Devlin, Keith. 2006. Situation theory and situation semantics. In *Handbook of the history of logic*, ed. Dov M. Gabbay and John Woods, volume 7, 601–664.

van Eijck, Jan, and Hans Kamp. 1997. Representing discourse in context. In *Handbook of logic and language*, ed. Johan van Benthem and Alice ter Meulen. The MIT Press.

Gawron, Jean Mark, and Stanley Peters. 1990. *Anaphora and quantification in situation semantics*. CSLI.

van Genabith, Josef, and Richard Crouch. 1999. How to glue a donkey to an f-structure or porting a dynamic meaning representation language into LFG's linear logic based glue language semantics. In *Computing meaning, volume 1*, ed. Harry Bunt and Reinhard Muskens, volume 73 of *Studies in Linguistics and Philosophy*, 129–148. Kluwer Academic Press.

Kamp, Hans. 1981. A theory of truth and semantic representation. In *Formal methods in the study of language*, ed. Jeroen Groenendijk, T.M.V. Janssen, and Martin Stokhof, 277–322. Mathematical Centre Tract 135, Amsterdam.

Kamp, Hans, and Uwe Reyle. 1993. *From discourse to logic*. Dordrecht: Kluwer.

Kohlhase, Michael, Susanna Kuschert, and Manfred Pinkal. 1996. A type-theoretic semantics for lambda-drt. In *Proc. of the 10th Amsterdam Colloquium*, ed. P. Dekker and M. Stokhof, 479–498. de Gruyter.

Kokkonidis, Miltiadis. 2005. Why glue your donkey to an f-structure when you can constrain and bind it instead? In *Proceedings of LFG05 Conference*, ed. Miriam Butt and Tracy Holloway King. CSLI Publications.

Lev, Iddo. 2006. On the syntax-semantics interface of overt and covert reciprocals. Paper for Ling223B, seminar on Quantification, Stanford University.

Lev, Iddo. 2007. Exact computational solutions to problems of natural language consequences. Dissertation draft. http://www.stanford.edu/∼iddolev/pulc/current_work.html.

Link, Godehard. 1998. *Algebraic semantics in language and philosophy*. Number 74 in CSLI Lecture Notes. CSLI Publications.

Muskens, Reinhard. 1995. *Meaning and partiality*. CSLI.

Muskens, Reinhard. 1996. Combining montague semantics and discourse representation. *Linguistics and Philosophy* 19:143–186.

Peters, Stanley, and Dag Westerståhl. 2006. *Quantifiers in language and logic*. Oxford University Press.

Reyle, Uwe. 1993. Dealing with ambiguities by underspecification: Construction, representation and deduction. *journal of semantics* 10:123–179.