

Direct Compositionality (and Variable-Free Semantics): The Case of Antecedent Contained Deletion

Pauline Jacobson
Dept. of (cognitive and) Linguistic Science(s)
Brown University
pauline_jacobson@brown.edu

May 25, 2001
UCLA Syntax/Semantics Seminar

I. The Basic Program and rationale

Direct compositionality (e.g., Montague, 1973)

combinatory syntactic operations builds expressions
(phrase structure rules + probably others, e.g., Wrap operations)
compositional semantics works "in tandem" - to directly supply model-theoretic
operation to each expression as it is built in the syntax

NOTE: model-theoretic interpretation: each expression assigned a meaning
in terms of "stuff in the universe" - truth values, individuals, possible
worlds, times, events(?), and combinations of these

consequences:

- no use in the grammar of an additional (syntactic) level of representation -
such as Logical Form (LF) which is then assigned a model-theoretic
interpretation (Formulas using logical-ish notation are just convenient
ways for the linguist to name model-theoretic objects; they have no status in
the grammar)
- no need for an additional set of rules mapping surface structures (or other
syntactic structures) into LFs
- no need for the syntax to work, in part, bottom up - and then the
semantics to "go back" and again compute meaning of the LF representation
bottom up

Sub-program: Variable-Free Semantics

Standard view:

- pronouns "translate" as variables at LF; variables at LF are assigned
meanings which are functions from assignment functions (ways to assign
values to variables) to individuals
- expressions have meaning only relative to "assignment functions" (i.e., relative
to ways to assign values to the variables)

Variable-free view:

- no variables, no assignment functions as part of the semantic machinery -
meanings are all "normal" healthy model-theoretic objects
- variables in the representations only - for notational convenience - the grammar
doesn't know anything about them

- no indexing in syntax

Overview and Goals:

- (1) examine the standard, classic argument for LF based on run-of-the-mill Antecedent Contained Deletion, and show that the domain is actually compatible with direct interpretation
 - in particular: there will be no need for "Quantifying In", "QR" or any other kind of Binders Out approach
 - note:* the classic argument for LF based on ACD is, in particular, an argument for the Binders Out aspect of LF
- (2) examine a slightly fancier argument for LF based on ACD and show that it too disappears once variable-free semantics is adopted
- (3) along the way, to show (briefly) that variable-free semantics has independent motivation
- (4) show that the semantics for Pied-Piping constructions comes automatically in Variable-Free semantics without any kind of "reconstruction" or earlier level at which the Pied-Piped material is in the position of the extraction site
- (5) develop a new case centering on interaction of ACD and Pied-Piping :
 - problematic for the standard view - under which ACD is crucially *VP Ellipsis*
 - but is not problematic for the analysis here - because ACD is "*TVP Ellipsis*"
 - all that's missing is a meaning of type $\langle e, \langle e, t \rangle \rangle$ - not a meaning of type $\langle e, t \rangle$

2. Background: VP Ellipsis in general

- (1) John will eat beans, and Bill will too.

most analyses adopt some combination of the following two cross-cutting parameters:

- (a) deletion (under some kind of identity with some other expression, see below) vs. interpretation (supply something)
- (b) identity of LF vs. identity of meaning

Note: Tancredi (1992), Rooth (1994) and others: in addition to identity, there is also a focus condition:
roughly: the ellipsis site must be contained in some constituent C such that the antecedent is contained in a constituent C' whose meaning (or something entailed or even implicated by the meaning of C') is an alternative to the meaning of C

e.g. - *John will eat beans* is an alternative to *Bill will (eat beans)*
and *Bill* must be focussed, hence "invoking" a set of alternatives

Moreover: recent attempts to get rid of identity condition altogether in favor of some extension of the focus condition (Tancredi, 1992, Fox, 1999, Merchant, 1999)

- Deletion + Identity of LF: Sag, 1976

- Interpretation + Identity of LF (= copy in an LF): Williams, 1977 and many others
- Deletion + Identity of Meaning: ??? (but compatible with approach here)
- Interpretation + Identity of Meaning (e.g., pick up a meaning): approach here (see also, Partee and Bach (1981), Hardt (1992), Dalrymple, Shieber, and Perriera (1992), etc.)

Assumptions here (some made purely for expository convenience):

(for more detailed treatment, see Jacobson, to appear in J. Kruiff and R. Oehrle (eds.), *Binding and Resource Sensitivity*)

- think of this like "free pronouns" - think of auxiliary as being like a pronoun (here, a pro-VP) (thus the term *VP Anaphora* is actually correct)
- for simplicity: we can think of this as encoding a "free variable"
- so *will* - basic item is a VP modifier
maps to second item includes a "free variable" - it is a VP proform and means will'(P)

(1) John will eat beans and Bill will too.

Bill will too ; b will'(P)

value for P gets contextually specified (like free pronouns), where it is made contextually salient by meaning of *eat beans*

Two obvious things to say more about:

- (i) observations in Hankamer and Sag (1976): unlike free pronouns, this is not an instance of "deep anaphora" - the ellipsis site meaning in general cannot be supplied purely by context, but must be supplied by overt linguistic material:

BUT: Since Schachter (1977): we know that it in fact can be supplied by context, especially if the context is "rich enough"

for various relevant examples: Schachter, 1977; Webber, 1978; Hardt, 1992; Dalrymple, Shieber and Pereira (1992) among others

the hope here: it is supplied by context, but the type of object that we need to supply (a pure function of type $\langle e, t \rangle$) is very fragile and not easily recovered just from context

- well known that it not only prefers a linguistic antecedent, but that it prefers a recent one
- hence, it very much likes to be supplied by being the meaning of some linguistically overt expression
- *thus: as is standard - this can be used (generally) as a diagnostic for the meaning of the antecedent*

questions for further research:

- *why* should this kind of object be "fragile" and difficult to pick up just from the context (or even from a meaning supplied in the non-recent discourse)?
- more specifically: this picks up something of type $\langle e, t \rangle$ but anaphora with *it* in "do it" construction is picking up something very close in meaning
- speculation: there's a slight difference between functions of type $\langle e, t \rangle$ and their "individual correlates" (cf., Chierchia, 1984) such that the latter are more stable, the former are not (but why??)

- note the similarity with the situation with paycheck pronouns

(ii) use of free variable: this is a cheat

- see Jacobson (to appear); this is just for expository convenience and is trivial to fix
- will; VP/RVP; will' ---> (maps by a rule which applies to all aux's)
will; VP^{VP}; will'
- hence, it's a kind of pro-form, and the information that anything containing it contains something unbound within it will be "passed up", so that
Bill will
will mean: P[will'(P)(b)] (as in the case of any other "free" pronoun)
- an interesting question is whether or not this is ever "bound" by the normal binding processes; see various literature on this

3. ACD: The received wisdom

3.1. Background: The received wisdom on relative clauses

(2) John will read every book which Mary will read.

axiomatic assumptions in lots of work in syntax and semantics

(i) the only method of semantic combination is by functional application

an even stronger assumption which is very standard:

(ii) all functions must be "saturated" - they must get all their arguments for semantic composition to proceed

• (i) follows from (ii); from (ii) also follows an additional thing (iii):

(iii) functions can't serve as arguments

the syntactic correlate:

all subcategorization desires of something must be satisfied (= "Projection Principle" - and hence empty PROs etc.)

assume: will' is a VP modifier (maps (time-dependent) sets into (time-dependent) sets

will'(leave') = a set of individuals x such that in the future leave'(x)

will' = P[x[F (P(x))]] (roughly)

informally: give me a property, and I'll give you back the set of individuals who will have that property in the future

(3) semantic composition of *every book which Mary will read*

will' needs to find a VP meaning

read' is a 2-place relation among individuals (not a VP-type meaning)

so: for the semantic composition to proceed: there must be a hidden object (a trace in the syntax and/or a level of representation in the syntax and a variable in the semantics) to give a VP-type meaning

read t read'(x)

(read'(x) has as its meaning a property relative to ways to assign values to
x)

will read t: $\text{will}'(\text{read}'(x))$
Mary will read t: $\frac{\text{will}'(\text{read}'(x))(m)}{m \text{ will}(\text{read}'(x))}$ or (for notational ease):

$m \text{ will}'(\text{read}'(x))$: a proposition relative to ways to assign values to \underline{x}
 this them mapped into a "closed property" by -abstraction --->

$\frac{x[m \text{ will}(\text{read}'(x))]}{}$
 = a "closed" property (on all ways to assign values to x , this is):
 the property of being an x such that Mary will read x

in set terms: $\{x\}m \text{ will read } x$

book which Mary will read: intersect above set with book' set

3.2. The consequences for ACD

(4) John will read every book which Mary will

- (a) as above: will' can only be happy if it finds a VP-type meaning (a set)
- (b) some kind of missing meaning is supplied (either deletion or interpretation)
- (c) missing meaning must be meaning of some (overt) linguistic expression (identity condition)
- (d) ergo: there has to be some linguistic expression at some level to supply a VP-type meaning

but: on surface: there is no VP which can supply the missing meaning
 only candidate is *read every book which Mary will*
 • but that can't be the one, since that leads to an infinite regress
 • i.e., that leads to the "antededent containment" paradox

LF to the rescue:

- posit a level of representation at which the object NP is "pulled out" of the clause, but where there is a trace/variable in object position ("Binders Out" approach)
- e.g., Quantifier Lowering: McCawley (1967), Lakoff (1971)
- Quantifying-In: Montague (1973) (technically not quite the same as positing LF, but does use a Binders Out level)
- Quantifier Raising: May (1976)

(5) for every book, x : [Mary will ____], [John will **read** x]

3.3 The fallacies here:

- no reason to assume only functional application

OR: *You can't (don't) always get what you want*

- Steedman (1987), Dowty (1989) and many others: can put relative clause meanings together directly, and without traces and/or variables in object position, by allowing function composition
 - informally - an incomplete expression like *read* - which hasn't found its object can directly combine with the auxiliary, and so on "up the tree"

(6) John will read every book which Mary will read.

let read' function compose with will'

read' is a function of type $\langle e, \langle e, t \rangle \rangle$

will' is looking for an $\langle e, t \rangle$, and thus is a function of type $\langle \langle e, t \rangle, \langle e, t \rangle \rangle$

will' o read' = $\lambda x[\text{will}'(\text{read}'(x))]$

this then function composes with the meaning of the subject

hence: assume that Mary' is a function from VP meanings to S meanings
 $P[P(m)]$ (i.e., the set of properties that the individual *m* has)

Note: This assumption is independent of the domain here; it is Montague's treatment of NP meanings in general, set up to give a meaning to quantified NPs; justified also for coordination of quantified and ordinary NPs

Mary' o $\lambda x[\text{will}'(\text{read}'(x))]$ = $P[P(m)] o \lambda x[\text{will}'(\text{read}'(x))]$ =

$\lambda y [P[P(m)](\lambda x[\text{will}'(\text{read}'(x))](y))]$ =

$\lambda y [P[P(m)](\text{will}'(\text{read}'(y)))]$ =

$\lambda y [\text{will}'(\text{read}'(y))(m)]$

for notational simplicity: $\lambda y [m \text{ will}'(\text{read}'(y))]$

- hence: final meaning same as in standard treatment
 - a set, which intersects with the head (ignore contribution of relative pronoun for now)
 - no need to posit a stage with an unbound variable in object position, which eventually gets bound
- note: have used a variable in the representation - but this treatment is *variable-free*
 - nothing ever has as its meaning a function which varies according to how we assign values to the variables
 - so we can strip away the assignment functions altogether

3.4. The consequences for the analysis of ACD

- Direct compositionality account of ACD (Cormack, 1984; Evans, 1989; Jacobson, 1992a, 1992b):
- the idea in a nutshell: the "missing meaning" here is just the 2-place relation read' (supplied by the meaning of the overt expression *read*) - and it combines up with the auxiliary just as it does in the full case

in more detail:

recall *will* as an anaphor: = will'(P)

(more technically, its meaning is $P[\text{will}'(P)]$ which = will'
 assume also: $\text{will}' \circ R$
 (that is, it is looking for a 2-place relation to compose with)
 (more technically, its meaning will be: $R[\text{will}' \circ R]$)

NOTE • This looks like a kludge - why the second *will*?
 • the beauty of the standard analysis is that ACD = VPE, and so nothing extra needs to be said

BUT: • once the full details of the system are supplied the same is true here; this second *will* (the one occurring in the ACD case) is automatic; nothing extra is needed

• most general form of the **g** ("Geach") rule will map:

AB to $(A/C)B/C$

• i.e., I contain a pro-form which has a "gap" of type C, and - modulo that - I am something with a gap of the same type

• hence: $\text{will}; VP^{VP}; \text{will}' \xrightarrow{\mathbf{g}} \text{will}; (VP/RNP)^{(VP/RNP)}$;
 $R[x[\text{will}'(R(x))]]$ which is equivalent to: $R[\text{will}' \circ R]$

in other words, this is nothing more than a pro-form over transitive verbs, rather a VP proform
 everything else proceeds in the usual way

• but, for expository simplicity, will think of *will* as looking for a 2-place relation to function compose with

(7) John will read everything which Mary will.

$\text{will}' \circ \text{read}'$ proceeds exactly as in (6)

again: in reality will' itself doesn't "pick up" read' - it just passes up the information that there is a missing TVP meaning, so that the final thing will end up being:

$R[j \text{ will read everything which Mary will } R]$
 and this is applied to read'

a related but different proposal: Lappin, 1995 - ACD as pseudo-gapping

3.5. Interlude:

Standard view: ACD is a special case of VP Ellipsis
 this view adopted here

Is TV(P) Ellipsis really same as VP Ellipsis?

• as with VP Ellipsis, it can happen across Ss in a discourse:

(8) a. VP Ellipsis: I ran. Bill did too.
 b. TVP Ellipsis: Bagels, I like. Donuts, I don't. (Evans, 1989)

NOTE: This is exactly the kind of case that Sag's condition on alphabetic variance was designed to preclude. However:

(i) his observations about the data were incorrect. One sort of case was:

(8c) Which book does Mary like? I don't know, which book does Sue?

Evans (1989) observes that this probably has something to do with the subject/aux inversion, note:

(8d) I know which book Mary likes, and I know which book Sue doesn't.

Additional cases which show that Sag's generalization was mistaken:

(8e) John_i asked Mary to water his_i plants, and Bill asked Sue to. (Jacobson, 1992; see also Hardt, 1992 for related cases) (sloppy reading okay)

(ii) Sag's definition of alphabetic variance was non-standard:

- Given two identical formulas F_1 and F_2 each of which contains within them an unbound occurrence of \underline{x} , they are alphabetic variants only if the two occurrences of \underline{x} are ultimately (higher up) bound by the same binder

Example: the elided VP and its antecedent in (8e) can both be represented as:

$x[x \text{ water } y\text{'s plants}]$

but the condition was defined to keep these from being alphabetic variants, since the two occurrences of \underline{y} are separately bound

- usual definition of alphabetic variance: two formulas are alphabetic variants iff they mean the same thing; here the two formulas do mean the same thing, and yet they are defined as non alphabetic variants - hence Sag's condition can't be stated compositionally (see Partee and Bach, 1981 for discussion) - have to look higher in the tree; and is non-standard

hence, it is a happy fact for everyone that the initial generalization was incorrect

- as with VP Ellipsis, ellipsis site can precede antecedent:

(9) a. VP Ellipsis case: Before John did, Bill climbed the mountain.
b. TVP Ellipsis: ? Everything which Bill does, I'll also read.

- as with VP Ellipsis, with enough work: can have a contextually salient but not linguistically overt antecedent:

(10) a. VP Ellipsis case: (i) I see you making salad, I pick up some carrots and a knife and say: Do you want me to? or Should I.
b. TVP Ellipsis case: Context: I see you about to reach for some cookies that just came out of the oven. Pointing first to a batch on a different plate, and then to the hot batch:

These, you may. Those you mustn't, at least not until they cool down.

- A question (for both the standard view and the view here):
 - all of these are harder to construct with TVP Ellipsis (or, in standard theory, with a VP containing a variable in object position) - *why?*

- tentative answer here: 2-place relations are even more "fragile" and difficult to access from context or from distant linguistic material than are meanings of type $\langle e, t \rangle$

3.5. TVP(!) ellipsis - not just simple transitive verb

(11) John thought that Bill would read everything that Sally also did (*think that Bill would read*)

- analysis has been developed to handle scope interaction facts in Sag (1976) - see Cormack (1984) and Jacobson (1992a)
- and various facts in Larson and May (1989), see Jacobson (1992a)
- plus additional interactions with Heavy NP Shift (Jacobson, 1992a)

4. But wait:

(12) John voted for every girl who thought he would. (*vote-for her*)

the apparent problem for the ACD as TVP Ellipsis analysis:

- what's missing here is a VP with a pronoun/variable in object position
vote-for'(her')

(13) a. John voted for every girl who thought he would vote for her.
b. *John voted for every girl who thought he would vote for.

- hence: need to supply vote-for'(x)
- in that case, we are back to the original argument for Binders Out level where the object is pulled out
- there is no surface VP whose meaning is vote-for'(x)
- but there is such a VP if we pull out the object

the fallacy: thinking that to do the semantic composition we need a *variable* in object position

5. Variable-Free semantics (which was motivated completely independently of this phenomenon)

5.1. The conventional way to think about pronominal binding

(14) Every man_i loves his_i mother.

(15) Every man_i thinks that he_i lost

A. The meaning of the pronoun and of expressions contain a pronoun unbound within them:

he, his ---> x

his mother: ---> x's mother (an "open" individual - i.e., an individual relative to ways to assign values to the variable)

he lost ---> λx *lost* (an "open" proposition; i.e., a proposition relative to ways to assign values to the variable)

B. How the pronoun eventually gets "bound": (Binders Out approach, though are others):

every man, x [x loves x 's mother]

λx *loves x 's mother*: an open proposition, depending on how assign values to x

this then mapped into a closed property (λ -abstract over x) --->

λx [*loves x 's mother*] (the set of x 's who love x 's mother)

this then taken as argument of *every man* (which is a set of sets)

5.2. Variable-Free view

NOTE: various implementations, cf.: Curry and Feys, 1958; Quine, 1965; Szabolcsi, 1989, 1991; Jacobson, 1991, ..., 1999 (*L&P*), 2000 (*NALS*), Dowty, 1992; Hepple, 1991; and various other works within CG)

- no variables or assignment functions as part of the semantic machinery
- pronoun: not a variable, but the identity function on individuals
- an expression containing a pronoun which is unbound within that expression: a function from individuals to something else (e.g., an "open" sentence: a function from individuals to propositions)
- binding a relationship between argument slots
- no co-indexation in syntax, since binding is not a relationship between two actual NPs/variables/traces - but between two argument slots

(16) Every man thinks that he lost:

he lost: λx [*lost*] = function from individuals to propositions; maps each individual into the proposition that that individual lost (= *lost*')

his mother: λx [*the-mother-of*'(x)] = function from each individual to that individual's mother (= *the-mother-of* function)

he: λx [x] = identity function on individuals

How get the effect of binding?

- type shift rule: \mathbf{z}

take a function wanting an argument of type a (an individual, proposition or whatever) and a higher individual argument (e.g., subject)
type shift it into something wanting as argument a function of type $\langle e, a \rangle$, and "merge" the newly created e -slot with its subject slot

(17) For any expression with meaning ' of type $\langle a, \langle e, b \rangle \rangle$ - this expression can map into a new meaning \mathbf{z} (') of type $\langle \langle e, a \rangle, \langle e, b \rangle \rangle$

where \mathbf{z} (') = f [λx ['($f(x)$)(x)]]

(18) Every man _{i} loves his _{i} mother

love' = is an ordinary relation between individuals

$z(\text{love}')$ - is a relationship between individuals and functions of type $\langle e, e \rangle$ such that to $z(\text{love}')$ some function f is to be an x who ordinary-loves $f(x)$

Every man loves his mother

his mother - the-mother-of function

to $z(\text{love})$ the mother of function is to be a self's mother-lover

i.e. $z(\text{loves}) \text{ his mother} = \lambda x [x \text{ loves the-mother-of} (x)]$

this then is a property (a set) and so it combines with *every man* just as in the conventional view

(19) *Every man believes that he lost*

believes: is a relation between individuals and proposition

$z(\text{believes}')$ wants a function of type $\langle e, t \rangle$ in object position

to $z(\text{believe}')$ a property P is to be an x who believes $P(x)$

he lost is of type $\langle e, t \rangle$

$z(\text{believe}) \text{ that he lost:} = \lambda x [x \text{ believes lost}'(x)]$

(occurs as argument of subject, exactly as in conventional account)

NOTE: *What is the cost? A type-shift rule.*

But anyone needs a type-shift rule. Standard theory also has one to effect binding - it's the λ -abstraction "rule". It's just a slightly different rule - and happens on a bigger domain (the meaning of the whole S) - whereas z can happen on a smaller domain - here, just the meaning of the verbs

How do we combine things with pronouns when binding isn't happening?

(e.g., why does *he lost* mean lost')?

lost' of type $\langle e, t \rangle$

it wants an individual in subject position

but he' is a function of type $\langle e, e \rangle$ (the identity function on individuals)

similarly: *his mother lost*

An additional type-shift rule: g (for Geach, 1974 "A Program for Logic", but also see a lot of CG literature here)

g maps a function of $\langle a, b \rangle$ into one of $\langle \langle e, a \rangle, \langle e, b \rangle \rangle$ (give me an incomplete a , and I'll pass up the information that I ultimately still need an e argument)

(20) For any expression with meaning ' of type $\langle a, b \rangle$, it can map into a corresponding expression $g(\text{ " })$ of type $\langle \langle e, a \rangle, \langle e, b \rangle \rangle$ with meaning

$f[\lambda x [\text{ '}(f(x))]]$

i.e., give me a function f , and an individual x , and I'll give you back what you would have gotten if you'd originally applied me to $f(x)$

(21) *he lost*: lost' of type $\langle e, t \rangle$ ---> $g(\text{lost}')$ of type $\langle \langle e, e \rangle, \langle e, t \rangle \rangle$ with meaning

$$f[\lambda x[\text{lost}'(f(x))]]$$

it combines with *he* which is the identity function, and what we get back is lost' (modulo gender information)

$$\text{he lost: } f[\lambda x[\text{lost}'(f(x))]](\lambda y[y]) = x[\text{lost}'(\lambda y[y](x))] = x[\text{lost}'(x)] = \text{lost}'$$

(22) *his mother*: will give the-mother-of function

53. Independent motivation for Variable-Free semantics

- theoretical motivation:
 - simplifies semantic machinery
 - variables arguably a "trick"
 - (arguably) more natural account of "free pronouns"
 - no LF/reconstruction needed
 - (traditional view: binding is a relationship between two NPs in a certain configuration. We often find a binder without a bindee, or a bindee without a bindee in the right configuration. So we move things around and/or add structure at LF to get the right configuration. Here, this is unnecessary)
- more empirical arguments:
 - (arguably) simplifies the analysis of: functional questions, answers to functional questions, unexpected inferences, unexpected connectivity effects in copular constructions, across-the-board binding and extraction, paycheck pronouns, i-within-i effects, interaction of paycheck pronouns with i-within-i effects, some unexpected exceptions to i-within-i effects, some unexpected exceptions to Weak Crossover effects, Pied-Piping

one example: functional questions (and their answers)

(23) Which woman does no Englishman love? His mother.

functional questions analysis: Groenendijk and Stokhof, 1983; Engdahl, 1986:

- "trace" or missing thing in object position can translate as a complex variable: a variable \underline{f} over functions of type $\langle e, e \rangle$ (from individuals to individuals) applied to an argument variable \underline{x} over individuals
 - hence "trace" here translates as $\underline{f}(\underline{x})$
 - \underline{f} is what is questioned
 - \underline{x} is bound by normal variable-binding:
- (24) what is the function f (from individuals to individuals, and with range woman') such that no-Englishman' (is an x) such that x loves $f(x)$?

here: nothing extra needs to be said - the existence of functional questions comes for free from the mechanisms for binding in general

hence: • no "complex meaning" for the gap

- as in any other case of extraction, gap is just a missing argument
(*loves* function composes with *no Englishman* instead of getting its object argument first)
- but, *loves* undergoes **z** - so that it is expecting an object of type <e,e>
- thus, what is missing is a simple argument of type <e,e>
(the argument position of that function is bound by the subject position of *love*)

(25) what is the function *f* such that no Englishman *z*-loves *f*?

this same as the G&S/Engdahl meaning, but the compositional semantics is different: we have just a "missing" functional argument - rather than a missing individual argument; the fact that we can be expecting a functional object is just part and parcel of the fact that we can have an object with a pronoun in it

the answers to functional questions:

(26) Which woman does no Englishman love? His mother.

his mother: standard story: denotes an (open) individual (*x*'s mother) i.e., an individual relative to ways to assign values to *x*

here: automatically denotes a function of type <e,e> = the-mother-of function

footnote: This argument doesn't go through if one has a theory in which the answer is actually a hidden *S* (and/or proposition).

But the same point can be made with cases where this is less appealing, as in, e.g.,
(27)

(27) The only woman that every Englishman loves is his mother.
(Geach, 1962; analysis relating these to functional questions in von Stechow, 1991 (see also Sharvit, 1999); simplification of von Stechow's analysis in which all the pieces come "for free" in Jacobson, 1994)

5.4. Implications for the ACD case

(28) John voted for every girl who thought (that) he would.

Interlude: dealing with the pronoun he

Note: these best with a pronoun in subject position (Haik, 1985)

(29) *John voted for every woman who thought (that) Bill would.

- claim: this has to do with focus and stress (not with the fact that we can have a bound pronoun):

(30) ?John voted for every woman who thought (that) John would. (at worst, a mild "Principle C" violation)

- (31) The Dating Game scenario: Imagine a t.v. show in which two men and two women are contestants each evening. Each woman writes down the name of one man that she would like to date. Each man writes down the name of a woman who he thinks will pick him. If the woman A picks the man B and B succeeds in guessing that A will pick B, then A and B get to go out on a fabulous date. But last night, no one one, because
- a. Unfortunately, Mary picked the man who thought that Sue would pick him, and Sue picked the man who thought that Mary would.
 - b. ?Unfortunately, Mary picked the man who thought that Sue would, and Sue picked the man who thought that Mary would.

to ignore irrelevant complication, will deal with (30)

- (30) John voted for every woman who thought that John would.

the solution:

- *vote-for her* means the same thing as *vote-for*
- i.e., only need to pick up the TVP meaning *vote-for'*
- the "object position" of this 2-place relation is bound in the same way that variable-binding happens in general
- in this case: *thought* undergoes **z** - so its subject slot "binds" the object position of *vote-for*

- (31) *would*; would' o R

R can be "filled in" as vote-for'

would' o vote-for' = x[would'(vote-for'(x))]

John - type lifted meaning, composes with this --->

x[j would'(vote-for-(x))] (type <e,t>)

thought; z(thought') = P[y[y thought P(y)]]

thought John would vote for; P[y[y thought P(y)]](x[j would'(vote-for-(x))])

= y[y thought (j would vote-for (y))]

this then intersects with woman' (set denoted by head)

- (32) *John voted for every woman who thought John would vote for

badness from the syntax, not the semantics

(for details, see Jacobson, to appear)

there is no way for the syntax to allow this - the story is exactly analogous to any version of a standard story

details (both in standard story and here) depend on precise syntax of subject *who* - assume it's an ordinary subject (not "extracted") so it wants to combine with a VP

i.e.: RC/R VP

- then it cannot combine with a transitive verb
- it can combine with a VP with a pro-form, by **g** - but that's not what it finds here
- it can combine with a VP with an extraction gap, but then that will be passed up and eventually has to be in a construction licensing that

further footnote: one does ultimately need to say what are the set of "standalone" categories

a remark: we would expect this to show the same properties as other cases of (T)VP "ellipsis":

-- across sentences in a discourse:

(33) John spoke to several girls. But Mary is the only one who wanted him to.

-- very difficult to get a "deep anaphora" case, but marginally possible with work:

(34) Context: (with apologies to the LSA regulations): a party with a bunch of junior high school students, and Johnny is going around trying to kiss every girl. I say to him: Hey, I think Sally is the only one who wants you to.

6. An advantage over the standard approach: Pied Piping and Ellipsis

6.1. Pied-piping without reconstruction

- the semantics of Pied-Piping is now automatic, making only one pretty reasonable assumption) without any kind of reconstruction
caveat: -- this is true for relative clauses
-- obviously, this needs to also work for questions
-- whether or not the extension to questions is automatic remains to be seen

(33) every man the father of whom Mary voted-for

- in functional questions, *vote-for* can undergo **z** to want a functional gap (whose argument is bound by the subject position of *vote for*)
- but it can also undergo **g** to want a functional gap whose argument position is not bound (yet), and is "passed up"
- another (simpler) derivation which gets the same result:
let ordinary *vote for* function compose with *Mary* (in the normal way)
 $\underline{P[P(m)]} \circ \underline{\text{vote-for}'} = \underline{x[m \text{ vote-for } x]}$
i.e., an $\langle e, t \rangle$ - or, a set
but this can undergo **g** so that it wants not an e-argument, but an $\langle e, e \rangle$ argument
i.e., it characterizes a set of functions of type $\langle e, e \rangle$ (rather than a set of individuals

$$\mathbf{g}(\underline{x[m \text{ vote-for } x]}) = \underline{f[y[x[m \text{ vote-for } x](f(y))]]} = \underline{f[y[m \text{ vote-for } f(y)]]}$$

informally: give me a function *f* (from individuals to individuals) and I'll give you the set of individuals *y* such that mary votes for *f(y)*

so, this wants a function as argument

the father of whom:

- assume that relative pronouns have the semantics of ordinary pronouns

- then, *the father of whom* is just the-father-of function
 - and so is of the right type to be argument here
- $$f[\ y[m \text{ vote for } f(y)]](\text{the-father-of}) =$$
- $$y[\ m \text{ vote for the-father-of}(y)]$$

A note of "semantic reconstruction": have shown out the λ -conversion step above, and in some literature this gets referred to as "semantic reconstruction"

- λ -conversion is only a notational device; the two formulas above are the same model-theoretic object; it's just two different ways to show and to notate the same thing
- hence no real sense in which the-father-of function is being "put back" into some syntactic representation

this now denotes a set, and so can intersect with the head

Note: a similar analysis proposed within the standard theory in Sharvit, 1998
 Sharvit also points out that given functional questions, can have a functional gap here
 but here, both parts come for free:

- the possibility of a functional gap comes from \mathbf{g} which is part and parcel of the mechanisms needed for variable-binding anyway
- the fact that *the father of whom* denotes a function is automatic from the semantics of pronouns

6.2. The payoff: the interaction of Pied-Piping with ACD

NOTE: assume the traditional view: there is an identity condition (on meaning and/or LF) with the antecedent

will return (if time) to consequences for views which dump this condition

(34) Mary voted for every candidate the FATHER of whom Bill had.

the problem: how can there be identity between the meaning (or, LF) of the matrix VP and the meaning (or, LF) of the ellipsis site?

- assume a reconstruction analysis of Pied-Piping
 - then the meaning of the matrix VP (assuming Binders Out) is:
 $\underline{\text{vote-for}(x)}$
 - the meaning of the ellided VP is $\underline{\text{vote-for}(\text{the-father-of}(x))}$
- assume something like Sharvit's analysis - where the gap is functional
 - then, the meaning of the matrix VP (assuming Binders Out) is:
 $\underline{\text{vote-for}(x)}$
 - the meaning of the ellided VP is $\underline{\text{vote-for}(f(x))}$

note: much discussion in traditional view about alphabetic variants
 one view: identity allows identity up to difference in variable names:

one version:
 $\text{vote-for}(x) = \text{vote-for}(y)$

another version:

$$x[x \text{ vote-for}'(x)] = y[y \text{ vote-for}'(y)]$$

the need to define "identical up to alphabetic variance" does not arise in Variable-Free semantics

variable names are an artefact (just convenient notational labels)

the fact that the above two formulas are equivalent is automatic, since they just name (the same) model-theoretic objects

but in any case: $\underline{f(x)}$ \underline{x} and so $\underline{\text{vote-for}'(f(x))}$ $\underline{\text{vote-for}'(x)}$

why the analogous problem does not arise in the account here:

- $\underline{\text{vote-for}'}$ is the meaning that we are picking up
- i.e., the TVP ellipsis analysis says identity is just identity of TVP meaning

a potential worry: Are we equating $\underline{g(\text{vote-for}')}$ with $\underline{\text{vote-for}'}$? and hence doing the same kind of cheat?

Answer: no. We can just "pick up" $\underline{\text{vote-for}'}$. The presence of the missing functional argument (which is filled by the meaning of *the father of whom*) comes about from \underline{g} on the whole thing - not \underline{g} on the "missing meaning"

(34) Mary voted for every candidate the FATHER of whom Bill had

had; $\underline{\text{had}' \circ R} = \underline{x[\text{had}'(R(x))]}$

pick up: $\underline{\text{vote-for}'}$ as value for R, so: $\underline{x[\text{had}'(\text{vote-for}'(x))]}$

NOTE: there really is no reason to think that this happens in the compositional semantics in this way

- in reality, the whole thing is a function from 2-place relations to the rest, and the value is supplied "at the end"
- however, the exposition is much simpler if we supply it right away

Bill had; $\underline{P[P(b)] \circ x[\text{had}'(\text{vote-for}(x))]} =$

$$\underline{x[P[P(b)](\text{had}'(\text{vote-for}(x))]} =$$

$$\underline{x[\text{had}'(\text{vote-for}'(x))(b)]}$$

notational ease: $\underline{x[b \text{ had}'(\text{voted-for}'(x))]}$

---> \underline{g} :

$$\underline{f[y[x[b \text{ had}'(\text{voted-for}'(x))](f(y))]]} =$$

$$\underline{f[y[b \text{ had-voted-for}'(f(y))]]}$$

i.e., give me a function and I'll give you back the set of y such that b voted for f(y)

this then gets *the-father-of* function and returns the set of y's such that b had voted for the father of y

crucially: we didn't apply **g** to the missing meaning, so we didn't skirt the identity condition

this works, because we don't need to pick up a full VP meaning, complete with a functional gap and an argument position for that gap (where there is no other VP to supply this meaning)

we need only pick up a TVP meaning; the functional argument position is supplied by the operation of **g** (which is part and parcel of the whole binding system)

6.3. is there a problem in the classical (VP Ellipsis) account of the identity condition is dropped?

- depends on what replaces it

Rooth's focus condition by itself is not enough - since deaccenting and ellipsis not the same:

(35) Mary called Bill an idiot, and then SALLY insulted him too.
(small font indicates de-accenting)

(36) Mary called Bill an idiot, and then SALLY did too.
does not mean: Sally insulted him too.

NOTE the following asymmetry:

(37) ??Every candidate the FATHER of whom BILL had, MARY voted for.
marginal, because ellipsis site not like to precede antecedent, but not awful

(38) *Every candidate the FATHER of whom BILL had voted for, MARY had.
impossible on the relevant reading

compare to:

(39) ??Every candidate who BILL had, MARY voted for. (also funny)

(40) Every candidate who BILL had voted for, MARY did. (fine)

explanation in Jacobson (1998): asymmetry from Rooth's focus condition:

alternatives to *the FATHER of whom Bill had* in (37):

- sets of individuals of the form $_y$ [a voted for f(y)]
which vary on the value of a (the voter) and f (the function)

meaning of *MARY had (voted for)* (after supplying the "missing meaning") =

- $_y$ [m voted for y]

hence, doesn't directly satisfy focus condition

but: implicational bridging: from this can derive

- $_y$ [m voted for id(y)]

that is: identity function can be the alternative here to the FATHER function above

- for independent evidence that identity function is a real alternative to more run-of-the-mill functions, see Jacobson (1998)

reverse does not work:

we can't get the meaning of *the FATHER of whom BILL had voted for* to be an alternative to

MARY had (voted for)

because there is no overt material to allow id to be in focus (and hence no overt material to send us off looking for an alternative to the identity function)