Functional Parasitic Gaps

Benjamin Russell
Brown University

1. Introduction

Munn (2001) observed that when a parasitic gap construction involves functional extraction as in (1a), there may only be a single binder for both the adjunct gap and the matrix VP gap, as in (1b). The reading with two binders paraphrased in (1c), however, is unavailable.

(1) a. Which relative did every boy hug _ after the president offended _ ?
b. for which f : every boy x hugged $f(x)$ after the president y offended $f(x)$
c. *for which f : every boy x hugged $f(x)$ after the president y offended $f(y)$

In an attempt to account for this fact, Munn made the following proposal:

(2) A parasitic gap is a null resumptive pronoun whose semantic type is e.

Munn's proposal is a stipulation about the semantic type of a particular unpronounced element, and Munn does not work out the details of how the incorporation of this stipulation into a compositional theory of parasitic gap constructions accounts for the distinction in (1a). In this paper, I examine existing theories of the compositional semantics of functional extraction and parasitic gap constructions. Correct theories of each phenomenon should work together to account for the phenomenon in (1a). Indeed, I find that variable-free theories of functional and parasitic extraction fit seamlessly together. And because the variable-free functional extraction theory assigns individual and functional gaps different semantic types, Munn's proposal (2) may be implemented directly by making a minor modification to the variable-free rule for parasitic gaps. In contrast, because ordinary gaps and functional gaps are both type e in variable-based theories, (2) must be modified to be a restriction on indices rather than types (the details are elaborated in section 3.3 below) to be incorporated into a standard variable-based theory of functional extraction. Moreover, the variable-based theory of parasitic gaps requires a non-compositional rule. Given this non-compositional rule, the appropriately modified (2) may not be incorporated into the standard variable-based functional extraction theories of Engdahl (1986) or Chierchia (1993), but instead requires Cresti's (1995) more complex formulation of Engdahl's theory. I conclude that the contrast between the simplicity, elegance, and naturalness of the variable-free theory and the relative intransigence of the variable-based theories of functional parasitic gaps provides a good reason to prefer the variable-free theory.

2. Variable-Free Functional Parasitic Gaps

2.1. *Extraction and Adjunction in Combinatory Categorial Grammar*

One of the motivators for the variable-free theory of anaphora developed in Jacobson (1999) is the variable-free theory of extraction developed in combinatory categorial grammar (CCG) (Steedman 1987). The variable-free theory of extraction preceded the variable-free theory of anaphora, perhaps because the standard variable semantics of traces is even less motivated than the variable semantics of pronouns, in two ways. First, pronouns certainly exist (they are natural language expressions), and so they must have some associated semantics. Traces are theory-internal elements; they have to exist because of the (largely unargued-for) theoretical assumption that all syntactic arguments must be saturated at all levels of representation. So traces only need a semantics at all if you assume they exist. Second, pronouns lead a double life: they may be free or bound. This is part of the appeal of the standard assignment function analysis, which captures this duality with a single pronoun semantics. But traces do not lead such a double life: they are always bound at the end of the derivation, so assignment functions are not as well suited to their analysis. CCG provides a syntax and semantics of extraction that does not depend on traces in the syntax, and does not, therefore, rely on variables in the semantics.

To ground the discussion of the variable-free theory of functional extraction and parasitic gaps, a brief presentation of a simple combinatory categorial grammar follows. The set CAT of syntactic categories may be defined recursively as follows:

(3) CAT is the smallest set such that:
   a. $\{S, N, NP, PP\} \subset CAT$, and
   b. if $\{X, Y\} \subset CAT$, then $\{X/Y, X\backslash Y, X^Y\} \subset CAT$.

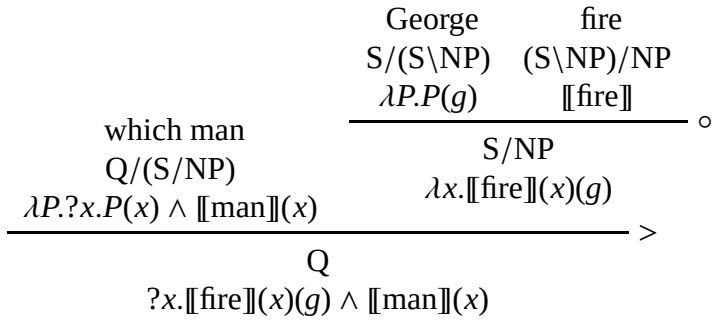
The grammaticality of a phrase is a derivation, with categories of lexical items as the premises and the category of the phrase as the conclusion. This view of the grammar contrasts with those in which grammars build *structure*; CCGs construct derivations, the internal structure of which is not "visible" to the grammar. In CCG derivations, the following steps are valid:

(4) a. **Forward and backward application**
   If X and Y are categories, then $X/Y\ Y \Rightarrow_> X$ and $Y\ X\backslash Y \Rightarrow_< X$, with corresponding semantics $f\ x \Rightarrow_> f(x)$ and $x\ f \Rightarrow_< f(x)$.
   b. **Forward composition**
   If X, Y, and Z are categories, then $X/Y\ Y/Z \Rightarrow_\circ X/Z$, with corresponding semantics $g\ f \Rightarrow_\circ g \circ f$.
   c. **Application of combinators**
   If α is a combinator, $X \Rightarrow_\alpha \alpha X$.

In CCG, argument slots are not always saturated—functions with unfilled argument slots may combine with other elements. Where standard theories posit an empty

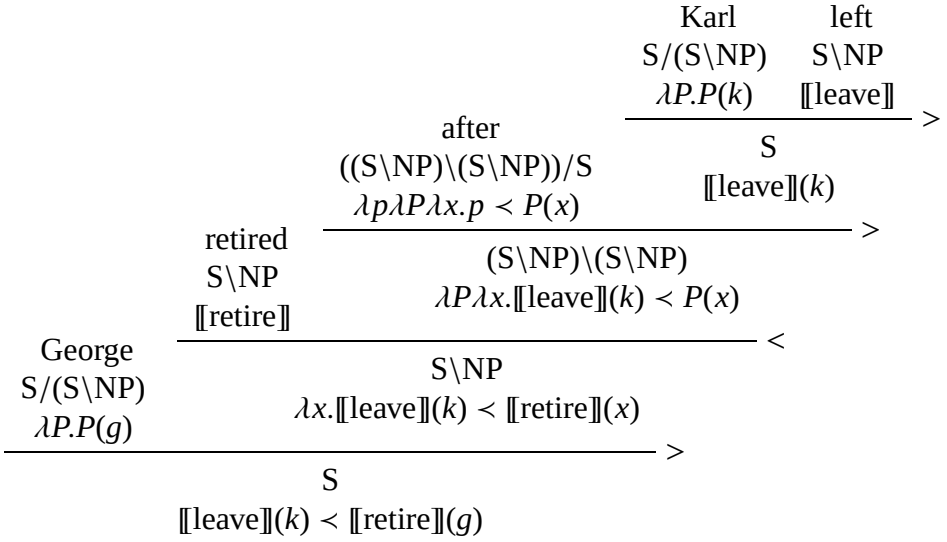
element—a trace—for extraction, function composition in CCG allows transitive verbs, for example, to function compose by (4b) with their subjects without receiving a direct object—in this way, extraction structures like (5) are built without positing any null elements.

(5) Which man did George fire?



Note further that there is no analog of Predicate Modification in this system. Adjuncts take the material they modify as arguments, so adjunction is just a special kind of function application, as illustrated in (6).

(6) George retired after Karl left.



Though this system was developed with somewhat different aims than to provide a variable-free analysis of extraction, it does indeed provide such an analysis. Furthermore, it has the desirable property of direct compositionality (Jacobson 2002): semantic and syntactic composition are simultaneous and inseparable.

2.2. *Variable-Free Parasitic Gaps*

Steedman (1987) developed one of the first fully compositional syntactic and semantic theories of parasitic gaps.[1] Steedman adapts the **S** combinator from the combinatory logic of Curry and Feys (1958) because its combinatory effect closely

mirrors the apparent semantic relation between the two gaps in parasitic gap constructions. In general terms, **S** allows a two-place function to temporarily forgo its first argument, then take as its second argument an element that would be the right type to be the second argument, except it's missing an argument of the same type as the argument that the original function has forgone. In the case of parasitic gaps, **S** allows gap-containing adjuncts to take gap-containing verb phrases as argument, merging the semantics of the two gaps.

(7) **Definition of S**[2]
S : $\langle$(A\B)/C, $f_{\langle e,\sigma\rangle}\rangle \mapsto \langle$(A/C)\(B/C),$\lambda g\lambda x[f(x)(g(x))]\rangle$

(8) Composition of a VP with parasitic gap using **S**:

$$\dfrac{\begin{array}{c}\text{filed}\\ \text{VP/NP}\\ [\![\text{filed}]\!]\end{array} \quad \dfrac{\begin{array}{c}\text{without reading}\\ \text{(VP\VP)/NP}\\ \lambda x\lambda P\lambda y[P(y)\wedge\neg[\![\text{read}]\!](x)(y)]\end{array}}{\begin{array}{c}\text{(VP/NP)\(VP/NP)}\\ \lambda Q\lambda x\lambda y[Q(x)(y)\wedge\neg[\![\text{read}]\!](x)(y)]\end{array}}\,\mathbf{S}}{\begin{array}{c}\text{VP/NP}\\ \lambda x\lambda y[[\![\text{filed}]\!](x)(y)\wedge\neg[\![\text{read}]\!](x)(y)]\end{array}}\,<$$

The application of **S** to an element of type $\langle\gamma,\langle\beta,\alpha\rangle\rangle$ allows it to forgo its first argument, the element of type γ. It is this argument that corresponds to the parasitic gap, so Munn's proposal is implemented in (7) by restricting the domain of **S** to functions where $\gamma = e$. This is a slight modification of Steedman's original definition, in which **S** only applies to elements whose first argument is an NP, with the aim of describing many of the same restrictions on parasitic gaps that Munn's proposal is meant to. This restatement of Steedman's restriction is, at present, no more or less stipulative than Steedman's original restriction. But, following Munn, the type e restriction may provide some hope of a deep semantic explanation of parasitic gap restrictions along the lines of Szabolcsi and Zwarts (1997).[3]

2.3. *Variable-Free Functional Gaps*

The standard theory of functional gaps is due to Engdahl (1986) and Groenendijk and Stokhof (1983). The key proposal in these works is that whereas a question like (9a) can be answered by naming an individual who is admired by all fiscal conservatives, questions like (9b) ask for a type $\langle e, e\rangle$ function (say, the function that maps a senator to his chief of staff).

(9) a. Which member of the government does every conservative admire most?
b. Which of his employees does every senator consult first?

Groenendijk and Stokhof's and Engdahl's original analyses of such questions depend on the use of variables in the semantics in which the argument of the function is bound by a quantifier, and the function is abstracted over to form a question

(details are in section 3.2 below). In Jacobson's (1999) adaptation of this theory, there are no variables: "binding" is achieved by the **z** combinator, an operation on two-place functions (like transitive verbs or transitive verb phrases; i.e., elements of category VP/NP).[4]

(10) **Definition of z**
$\mathbf{z} : \langle (\mathrm{A}\backslash\mathrm{B})/\mathrm{C}, f\rangle \mapsto \langle (\mathrm{A}\backslash\mathrm{B})/\mathrm{C}^{\mathrm{B}}, \lambda g \lambda h[f(g(h))(h)]\rangle$

(11) Example of "binding" with **z**:

$$\dfrac{\text{Every boy}\quad \dfrac{\dfrac{\dfrac{\text{loves}}{(\mathrm{S}\backslash\mathrm{NP})/\mathrm{NP}}}{(\mathrm{S}\backslash\mathrm{NP})/\mathrm{NP}^{\mathrm{NP}}}\,\mathbf{z} \qquad \dfrac{\text{his mother}}{\mathrm{NP}^{\mathrm{NP}}}}{\mathrm{S}\backslash\mathrm{NP}}\,>}{\mathrm{S}}\,>$$

Every boy	loves		his mother	
S/(S\NP)	(S\NP)/NP		$\mathrm{NP}^{\mathrm{NP}}$	
$\lambda P.boy \subseteq P$	⟦loves⟧	**z**	⟦the-mother-of⟧	
	$\lambda f \lambda x[[\![\text{loves}]\!](f(x))(x)]$			
	$\lambda x[[\![\text{loves}]\!]([\![\text{the-mother-of}]\!](x))(x)]$			>
$boy \subseteq \lambda x[[\![\text{loves}]\!]([\![\text{the-mother-of}]\!](x))(x)]$				>

z applies to *love*, allowing it to take a type $\langle e, e\rangle$ direct object, and *his mother* is a type $\langle e, e\rangle$ function (the function that maps individuals to their mothers). **z**-love identifies the semantics of the argument of *his mother* with the semantics of **z**-*love*'s second argument; i.e., its subject. This identification of argument slots has the interpretive effect of "binding" the argument of the functional direct object to the subject.

Because functions in CCG do not always get their arguments, and **z** allows transitive verbs to take functional arguments, functional questions like (9b) may also be built by **z**: for example, **z**-*love* may simply not receive its object, as in (12).

(12) The functional reading of the question "Who does every boy love?"

$$\dfrac{\dfrac{\text{who}}{\mathrm{Q}^{\mathrm{NP}}/(\mathrm{S}/\mathrm{NP}^{\mathrm{NP}})}\quad \dfrac{\text{every boy}\quad \dfrac{\dfrac{\text{loves}}{(\mathrm{S}\backslash\mathrm{NP})/\mathrm{NP}}}{(\mathrm{S}\backslash\mathrm{NP})/\mathrm{NP}^{\mathrm{NP}}}\,\mathbf{z}}{\mathrm{S}/\mathrm{NP}^{\mathrm{NP}}}\,\circ}{\mathrm{Q}^{\mathrm{NP}}}$$

who	every boy	loves		
$\mathrm{Q}^{\mathrm{NP}}/(\mathrm{S}/\mathrm{NP}^{\mathrm{NP}})$	S/(S\NP)	(S\NP)/NP		
$\lambda P.?f.P(f)$	$\lambda P.[[\![\text{boy}]\!] \subseteq P$	⟦loves⟧	**z**	
		$\lambda f \lambda x[[\![\text{loves}]\!](f(x))(x)]$		
	$\lambda f.[[\![\text{boy}]\!] \subseteq \lambda x[[\![\text{loves}]\!](f(x))(x)]$			∘
$?f.[[\![\text{boy}]\!] \subseteq \lambda x[[\![\text{loves}]\!](f(x))(x)]$				

As illustrated, there is no need for a downstairs argument variable. So the functional gap, in variable-free semantics, is type $\langle e, e\rangle$, not type e.[5]

2.4. *Interaction of* ***S*** *and* **z**

Putting everything together, **S** allows gap-containing adjuncts to combine with transitive verb phrases, yielding new transitive verb phrases containing parasitic gaps. **z** applies to transitive verb phrases and changes them into transitive verb phrases that take functional objects, binding argument slots. In terms of types, **z** shifts an element of type $\langle e, \langle e, t\rangle\rangle$ to type $\langle\langle e, e\rangle, \langle e, t\rangle\rangle$, and **S** shifts an element from type $\langle e, \langle\langle e, t\rangle, \langle e, t\rangle\rangle\rangle$ to type $\langle\langle e, \langle e, t\rangle\rangle, \langle e, \langle e, t\rangle\rangle\rangle$, the type of a parasitic adjunct; this can then take a type $\langle e, \langle e, t\rangle\rangle$ (VP with direct object gap) argument to yield an $\langle e, \langle e, t\rangle\rangle$ result. Notice that the output of **S** is the right type to build an input to **z**, but the output of **z** is not the right type to build an input to **S**. In other words, the output of **z** contains a functional gap, and due to the incorporation of Munn's proposal into the definition of **S**, items with functional gaps are not in the domain of **S**. This asymmetry accounts for Munn's generalization. The single-binder reading may be composed as in (13) by applying **S** first, then **z**. But to obtain a two-binders reading, **z** would have to apply to each transitive verb individually, before **S** applies, so that each verb's subject is a "binder". And this would yield elements that cannot be composed by **S** into a parasitic gap construction, thereby ruling out the two-binders reading, as illustrated in the failed derivation in (14).

(13) The single-binder derivation:
for which f : ⟦every boy⟧(λy[⟦hug⟧($f(y)$)(y) ≻ ⟦offend⟧($f(y)$)($\iota prez$)])

		after the president offended	
		(VP\VP)/NP	
		$\lambda x\lambda P\lambda y[P(y) \succ$ ⟦offend⟧$(x)(\iota prez)]$	**S**
hug		(VP/NP)\(VP/NP)	
VP/NP		$\lambda Q\lambda x\lambda y[Q(x)(y) \succ$ ⟦offend⟧$(x)(\iota prez)]$	
⟦hug⟧			<
	VP/NP		
	$\lambda x\lambda y[$⟦hug⟧$(x)(y) \succ$ ⟦offend⟧$(x)(\iota prez)]$		
			z
	VP/NP^{NP}		
	$\lambda f\lambda y[$⟦hug⟧$(f(y))(y) \succ$ ⟦offend⟧$(f(y))(\iota prez)]$		

(14) Failed two-binders derivation:
for which f : ⟦every boy⟧(λy[⟦hug⟧($f(y)$)(y) ≻ ⟦offend⟧($f(\iota prez)$)($\iota prez$)])

			offended	
			VP/NP	
			⟦offended⟧	**z**
after	the president		VP/NP^{NP}	
(VP\VP)/S	S/VP		$\lambda f\lambda x[$⟦offended⟧$(f(x))(x)]$	
⟦after⟧	$\lambda P.P(\iota prez)$			∘
		(VP\VP)/NP^{NP}		
		$\lambda f\lambda P\lambda x[P(x) \succ$ ⟦offended⟧$(f(\iota prez))(\iota prez)]$		
				***S**

To summarize, the definition of **S** in (7) incorporates Munn's proposal (23) by restricting application of **S** to functions whose first argument is type *e*. This restriction interacts with **z**, preventing **S** from applying to elements that have undergone **z** and predicting that functional parasitic gaps will have single-binder readings as in (1b), but not two-binder readings as in (1c). Munn's generalization is a very natural result in the variable-free system; the only necessary modification is the restatement of Steedman's syntactic NP restriction on the **S** combinator as a semantic type *e* restriction. So the empirical phenomenon of functional parasitic gaps is easily "handled" in variable-free semantics, setting the stage for furture research that might actually explain what here, and in Munn, has been stipulated.

3. Variable-Based Theories of Parasitic and Functional Gaps

This section evaluates what I will call "standard" semantics for gaps, functional gaps, and parasitic gaps. The standard semantics for ordinary gaps is the one presented in Heim and Kratzer (1998), the standard for functional gaps will be the theory developed in Engdahl (1986) and Chierchia (1993), and for parasitic gaps, the standard theory is the one developed in Nissenbaum (2000). For the purposes of this paper, what distinguishes these theories from the variable-free theory is, obviously but crucially, that they use variables in the semantics for extraction. This leads to a fundamental distinction between theories in the semantic type of gap-containing elements. For example, in CCG, an expression containing a individual-type direct object gap like "George bought" is type $\langle e, t\rangle$; in the standard variable-based theories, the same expression is type t. Because of this, expressions that contain gaps have different combinatory potential in the two types of theories. In particular, in the standard theory, VPs with gaps need to be shifted to type $\langle e, t\rangle$ in order to combine with parasitic adjuncts, and the rule Nissenbaum develops to accomplish this is (apparently necessarily) non-compositional (the details are below). Second, because Nissenbaum proposes that VPs with gaps combine with parasitic adjuncts through Predicate Modification, his rule can not provide for the composition of grammatical single-binder functional parasitic gaps: an adjunct containing a type *e* parasitic gap is type $\langle e, t\rangle$, whereas a VP with a type $\langle e, e\rangle$ functional gap is type $\langle\langle e, e\rangle, t\rangle$. These types cannot combine, making a wrong prediction of ungrammaticality in the case of single-binder readings of functional parasitic gaps. To remedy this, Cresti's (1995) theory of functional extraction must be adopted.

3.1. *Nissenbaum's Theory of Parasitic Gaps*

Parasitic gaps, much discussed in the syntactic literature (see Culicover and Postal (2001) and references therein), have rarely been addressed in the semantic literature. In fact, standard syntactic analyses proposed for parasitic gaps present a considerable challenge for any semantic theory that aims to account for their compositional interpretation. Nissenbaum (2000) develops a system for parasitic gaps within the

following set of assumptions.

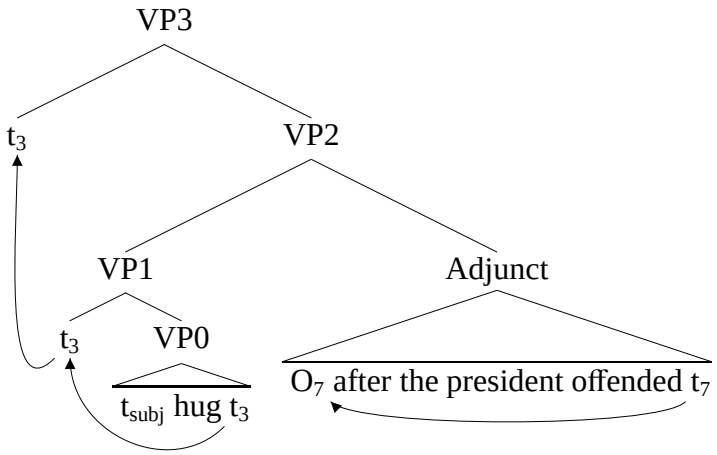
(15) a. Parasitic gaps are not in a chain with their licensing gaps (Chomsky 1986).
b. A parasitic adjunct and its licensing VP, which has an internal subject, are sisters.
c. Movement leaves traces which are interpreted compositionally as variables.
d. Parasitic adjuncts involve the movement of a null operator; syntactically they are similar to relative clauses.

In addition to these assumptions (none of which are unprecedented), Nissenbaum argues for the following grammatical principle.

(16) "Successive-cyclic A-bar movement targets a specifier position of every vP along the way to the final landing site."

(15) and (16) together give the syntax in (17) (VPs are numbered for easy reference, not because they are indexed or of different categories).[6]

(17)



Existing rules cannot provide for the interpretation of the tree in (17). This is, quite simply, because there is a type mismatch between the sisters labeled VP1 and Adjunct. (Nissenbaum assumes an event semantics in which an ordinary non-gap-containing adjunct is type $\langle \epsilon, t \rangle$, where ϵ is the type of events. His analysis does not, however, depend critically on event semantics; what is critical is that VPs have the same semantic type as adjuncts—this, of course, allows them to combine by Predicate Modification. So for the purposes of this paper, since nothing hinges on it, the added complexity of this event argument is glossed over: I treat adjuncts as if they are extensionally type t.) The semantics of parasitic adjuncts are composed the same way as relative clauses, giving a parasitic adjunct the extension in (18).

(18) $[\![\text{after the president offended}]\!] = \lambda x_k. \succ [\![\text{offended}]\!])(x_k)(\iota prez)$

The adjunct is type $\langle e, t \rangle$; to compose it with VP1, VP1 must also be $\langle e, t \rangle$. But, using ordinary Predicate Abstraction, the meaning of VP1 in (17) with respect

to an assignment g is $[\![\text{hug}]\!](g(x_3))(g(x_{subj}))$, which is type t. Ordinary Predicate Abstraction can not build abstracted predicates that have an independent "life" because abstracted predicates always combine with the element associated with the trace that induced the abstraction in the first place. In (17), for example, the trace t_3 does induce lambda abstraction over VP0, shifting the semantics of VP0 to $\lambda x.[\![\text{hug}]\!](x)(g(x_{subj}))$, which has the right semantics to be modified by the adjunct semantics in (18). But the predicate instead must combine with the trace responsible for inducing abstraction, freeing the variable x_3, and resulting in a VP1 that is type t and cannot combine with the $\langle e, t\rangle$ adjunct because of a type mismatch.

To ameliorate this type mismatch, Nissenbaum introduces the semantic rule in (19), which subsumes ordinary Predicate Abstraction.

(19) **Parasitic Predicate Abstraction**

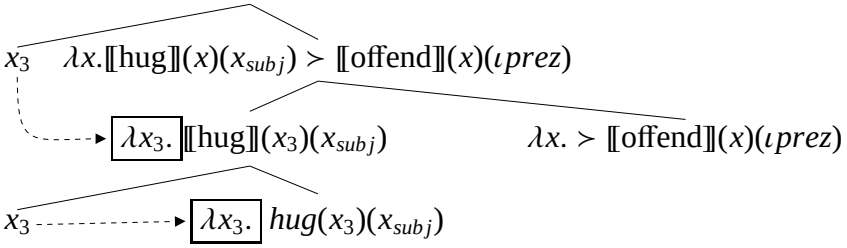
If γ is the head of a chain with index j, let β designate its sister. Then:

i. If β has a semantic value A determined by function application, then shift A to $\lambda x_j.A$.

ii. Otherwise, let β now designate the projecting daughter of the node referred to as β in step i., and return to step i.

This rule allows moved elements to induce abstraction lower than their sister nodes; that is, it dissociates moved elements from the predicates they create through abstraction. In particular, if a type t VP has as its sister an $\langle e, t\rangle$ gap-containing adjunct, composition is delayed until a suitable element higher in the tree induces abstraction over the VP. In the tree in (17), then, the t_3 in the Spec of VP3 induces lambda abstraction over x_3 on VP1. The semantic composition for the LF in (17) is spelled out in (20). Notice that the lower abstraction is ordinary Predicate Abstraction, which corresponds to just clause i. of (19). The higher abstraction is (19)'s innovation: this allows abstraction to be divorced structurally from the trace that triggers it, giving the abstracted predicate a "life" of its own, which allows it to combine with the adjunct predicate.

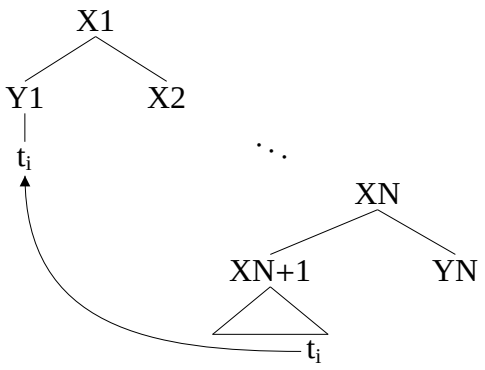
(20) $[\![\text{hug}]\!](x_3)(x_{subj}) \succ [\![\text{offend}]\!](x_3)(\iota prez)$



The dotted arrows point to the effects of (parasitic) predicate abstraction: the boxed lambdas are contributed by the traces found at the tail ends of their arrows. Notice that VP2 does not have a semantic value independent of VP3, and so Nissenbaum's rule is not compositional, since not every expression's meaning depends on only the meanings of its constituents.[7] This problem is rather more dramatic when viewed in more abstract terms: Let YN and XN + 1 be constituents with semantic type $\langle \sigma, \tau\rangle$

and τ, respectively, and consider the LF below for arbitrarily large N.

(21)



The subtrees dominated by X2 through XN do not receive a local interpretation. When the subtree dominated by X2 combines with Y1, the trace that has landed at Y1 triggers Parasitic Predicate Abstraction way down the tree on XN+1. Then XN+1 may combine with YN by Predicate Modification, and the meanings for each intermediate subtree may be computed. This means the meaning of a constituent may depend on an a constituent arbitrarily higher in the tree, a gross violation of the principle of compositionality.

But assuming that Nissenbaum's syntax is correct, the only available semantic composition rule for parasitic gap constructions is a non-compositional one. The non-compositional nature of Nissenbaum's analysis may appear to the reader to be an artefact of his way of stating the predicate abstraction rule. Others, including Heim and Kratzer (1998), propose that movement inserts an index in a tree, and this index is interpreted as a lambda abstraction operation. Thus one could imagine a restatement of Nissenbaum's system where movement causes a lambda-abstracting index to be inserted in the right spot, and that element is interpreted entirely compositionally. The problem with this approach (and perhaps the reason why Nissenbaum didn't take it) is that it is apparently not possible to find the right spot to insert the lambda abstraction operator without appealing to semantics—that is, Parasitic Predicate Abstraction abstracts over just the node whose semantic interpretation depends on it. A syntactic operation on LFs, *à la* Heim and Kratzer, can not be sensitive to such information, and therefore the right insertion rule can not be defined.

In this section I've presented suggestive (though not conclusive) evidence that the relatively uncontroversial assumptions made by Nissenbaum preclude a compositional analysis of parasitic gap constructions. In what follows, I will nonetheless assume that Nissenbaum's non-compositional rule is the right one for a variable-based theory of parasitic gaps in order to examine the interaction of this theory with variable-based theories of functional extraction.

3.2. *Engdahl's Theory of Functional Gaps and Chierchia's Implementation*

Engdahl proposed that whereas ordinary questions like (9a) are built by abstracting over individual-type variables, functional questions like (9b) are built by abstracting over higher $\langle e, e\rangle$-type variables (for more complicated functional questions, variables of type $\langle e, \ldots, \langle e, e\rangle \cdots \rangle$ are necessary, but this paper will be restricted to discussion of type $\langle e, e\rangle$ functional variables). To make formulas a little more readable, type e variables will be written with odd-numbered indices on the letter x, and $\langle e, e\rangle$ variables will have even-numbered indices on the letter f.

A modern implementation of Engdahl's theory is due to Chierchia (1993), which gives the system for functional extraction that follows. A trace may be a collection of variables, or, in other words, a trace may have multiple indices. This means that the interpretation of a single trace may depend on the values assigned to more than one variable. In particular, a trace may contain the index of a functional variable, and an index for that function's argument(s). So the LF for (9b) contains the subtree [every senator$_3$ [t$_3$ [consult t$_{3,6}$]]]. The extension of this bit of LF with respect to an assignment g is $[\![\text{every}]\!]([\![\text{senator}]\!])(\lambda x_3.[\![\text{consult}]\!](g(f_6)(x_3))(x_3))$. Note f_6 is free in this formula, and x_3 is bound. Now, to form a functional predicate, Chierchia has the movement of the doubly-indexed element $t_{3,6}$ induce abstraction over the index of its highest type. So predicate abstraction can be formulated as follows:

(22) **Functional Predicate Abstraction**
If α has daughters β and γ, and β is the head of a chain with indices $a_1, \ldots, a_n$, let a_i be the index associated with the highest-typed variable.[8] Then shift γ to $\lambda\chi_{a_i}.\gamma$, and combine this with β by function application.

Notice that this formulation of predicate abstraction imposes a requirement on the head of a multiply-indexed chain: it must be functional in type. That is, whereas the trace left by the extraction of a functional element is type e (the function variable applied to argument variables), the moved material is functional type. Further, intermediate traces must also be type $\langle e, e\rangle$, since they must combine with an element that is abstracted over a functional index. This means that the various traces in a chain have different semantics: the downstairs trace is multiply-indexed, but the intermediate traces are all singly-indexed (and, in particular, they share the index of the head of the chain, which is a type $\langle e, e\rangle$ index). This is the only way to make the semantics work out correctly given Chierchia's assumption about the semantics of functional traces and the assumption that movement leaves intermediate traces.

To summarize so far: in the Engdahl/Chierchia system, downstairs functional traces are type e elements with multiple indices. Intermediate functional traces are simply functional $\langle e, e\rangle$-type variables.

3.3. *An Attempt to Incorporate Munn's Proposal in the Variable-Based Theories*

Munn's proposal from (2) is repeated in (23).

(23) Parasitic gaps are of type *e*.

In Engdahl's system, functional traces are type *e*: a type $\langle e, e\rangle$ function applied to an individual variable argument. So (23), in fact, does not prevent parasitic gap traces from having functional indices. A quantifier within a parasitic adjunct may therefore bind the argument index, and two-binder readings are not ruled out. So it seems Munn's proposal must be rewritten to refer to indices, as in (24).

(24) Parasitic gaps are singly-indexed variables of type *e*.

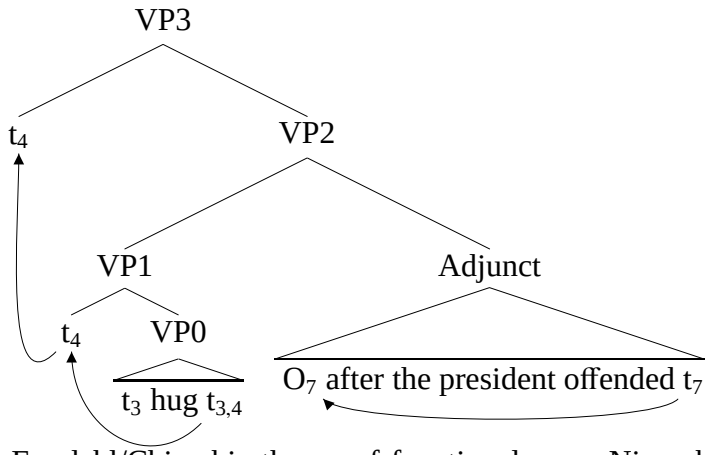
No overt pronouns seem to be subject to this kind of restriction; they all seem to have available paycheck (i.e., multiply-indexed) readings (Jacobson 2000). Further, resumptive pronouns seem to be no exception to this generalization; (25) has a paycheck, or functional, reading.

(25) Which of his relatives does every little boy wonder whether she loves him?

Moreover, reference to indices rather than types is not faithful to Munn's main idea, which is that it is the type, not number of indices, of a parasitic gap that is resposible for the no-two-binders generalization.

Nonetheless, having been suitably modified, it seems (24) accurately rules out ungrammatical two-binders functional parasitic gap constructions like (1c). Effectively, (24) restricts parasitic adjuncts to type $\langle e, t\rangle$—functional parasitic gap constructions may only have LFs where the parasitic gap is singly-indexed, as in (26). And since there cannot be a functional index in the parasitic adjunct, no quantifier inside the adjunct can bind an argument of a functional trace.

(26)



But the Engdahl/Chierchia theory of functional gaps, Nissenbaum's Parasitic Predicate Abstraction rule for parasitic gaps, and (24) cannot simultaneously be right—they cannot compose ordinary single-binder functional parasitic gaps like (1b), which are perfectly grammatical. To see this, the obvious way to combine Functional and Parasitic Predicate Abstraction is in (27).

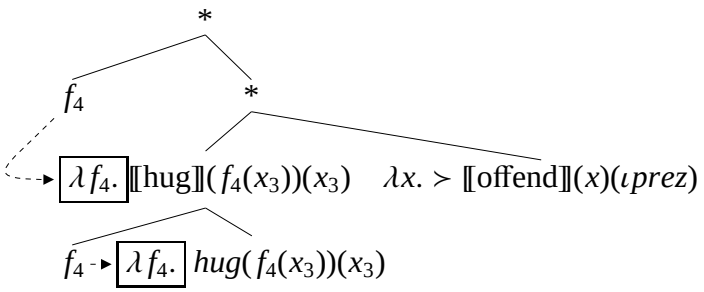
(27) **Functional Parasitic Predicate Abstraction** If β is the head of a chain with indices $a_1, \ldots, a_n$, let a_i be the index associated with the highest-typed variable, and let γ designate β's sister. Then:

i. If γ has a semantic value A determined by function application, then shift A to $\lambda\chi_{a_i}.A$

ii. Otherwise, let γ now designate the projecting daughter of the node referred to as γ in step i., and return to step i.

The semantics of VP1 with respect to an assignment g before application of Functional Parasitic Predicate Abstraction is $[\![\text{hug}]\!](g(f_4)(g(x_5)))(x_5)$; this rule induces abstraction over f_4, giving, for any assignment g, $\lambda f.[\![\text{hug}]\!](f(g(x_5)))(x_5)$. Like other parasitic gap constructions, VP1 should combine with the parasitic adjunct through Predicate Modification. But there is a type mismatch, as shown in (28), and so the LF in (26) is uninterpretable.

(28)

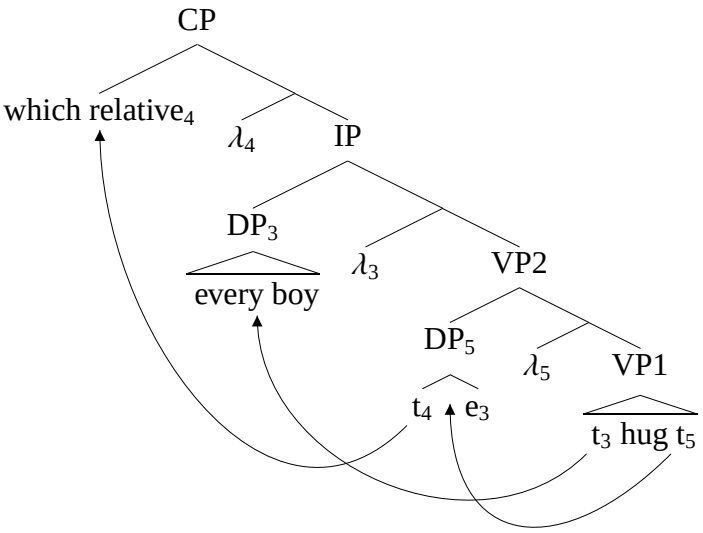


The parasitic adjunct is an ordinary $\langle e, t\rangle$ (suppressing the event argument) predicate, whereas the parasitic-abstracted VP is type $\langle\langle e, e\rangle, t\rangle$. These cannot combine through Predicate Modification or any other available rule, so legal one-binder parasitic gaps are incorrectly ruled out by the incorporation of Munn's proposal into Engdahl/Chierchia/Nissenbaum. As it stands, then, given Munn's modified proposal, Nissenbaum's theory of parasitic gaps is incompatible with Engdahl/Chierchia's of functional extraction.

3.4. *Cresti's Structured Traces*

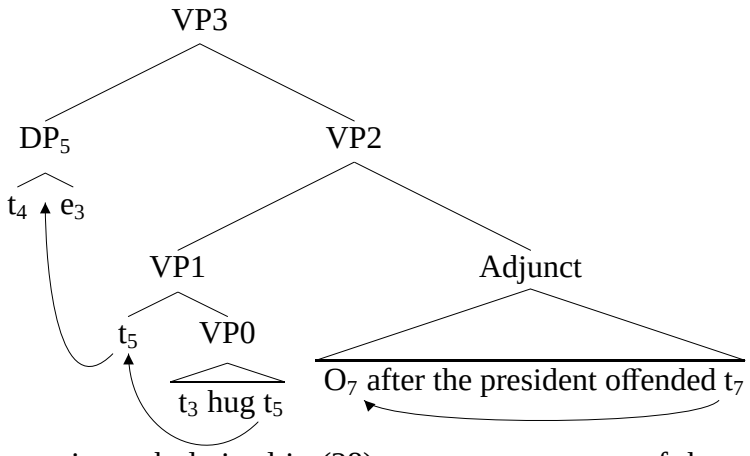
Until now, I've assumed functional traces have the structure and indices that Chierchia (1993) does: a trace may have a functional index and indices for that function's arguments. But Cresti (1995) argues that functional traces have a more complicated system of indexing in which there are additional *result* indices. In this system, LFs like (29) may be generated.

(29)



Notice that the functional trace is a complex DP which can move *in toto*, leaving a singly-indexed trace t_5 and inducing Predicate Abstraction over an individual-type index. This system, combined with Nissenbaum's syntax for parasitic gaps, can build the following functional parasitic gap LF:

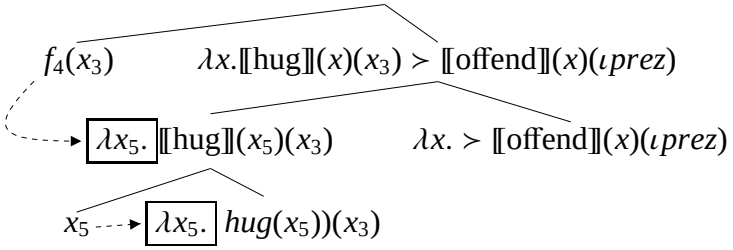
(30)



The type mismatch derived in (28) was a consequence of the assumption, following Chierchia (1993), that the movement of a functional element triggers lambda abstraction over the index of the functional variable. But in Cresti's system, the movement of a complex functional DP induces Predicate Abstraction over the index of the entire DP, which is type *e*. Then the functional element can move out of the structured functional trace, abstracting over the functional index.

Cresti's functional traces, rather than Chierchia's, combined with Nissenbaum's Parasitic Predicate Abstraction, make it possible to compose functional-gap-containing VPs with parasitic adjuncts, as in (31).[9]

(31) $[\![\text{hug}]\!](f_4(x_3))(x_3) \succ [\![\text{offend}]\!](f_4(x_3))(\iota prez)$



But Cresti's proposal, though necessary for the proper composition of functional parasitic gap constructions, is otherwise questionable, for at least two reasons. First, movement out of a DP as in (29) is generally syntactically prohibited. So Cresti's proposed syntax for functional elements requires an unprecedented and peculiar operation, reserved just for functional elements.

Second, Cresti's proposal that such structures exist is not independently justified. Her proposal is based on the empirical claim that (32a) is ungrammatical, in contrast to (32b) (indices only indicate relevant readings).

(32) a. *Which of his_i relatives_j would no senator buy Andres Serrano's portrait of $__{}_{i,j}$?
b. Which of his_i relatives_j would no senator buy a grotesque portrait of $__{}_{i,j}$?
c. *No senator_i would buy Andres Serrano's portrait of himself_i.
(Cresti's grammaticality judgments)

To explain this, Cresti proposes that the argument indices in functional traces are grammatically represented as null reflexive pronouns. This means functional traces must obey Principle A—the null reflexive pronoun must move close enough to a suitable quantifier (or a trace thereof) to be locally bound by it. For this to be possible, the entire complex functional trace must move. In (32b), this functional trace complex moves out of the indefinite NP *a grotesque portrait of* __, allowing the null reflexive to be bound by the subject *no senator*. But in (32a), the functional trace cannot move out of the definite NP *Andres Serrano's portrait of* __, incurring a Principle A violation like (32c). In sum, then, Cresti's empirical evidence for her structured traces is the purported contrast between (32a) and (32b).

Contrary to Cresti, my judgment (and the judgment of my informants) is that (32a) is considerably less degraded than (32c) and is about as good as (32b), especially if (32a) is asked in a pragmatically natural context: suppose Andres Serrano, the contemporary artist responsible for the creation of the work *Piss Christ*, starts doing portraits of senators' family members. Then (32a) seems fine, and "His mother, for one" is a reasonable answer. And if (32a) is, in fact, grammatical, Cresti's argument for anaphoric elements within functional traces disappears, and so Cresti provides no reason to think that complex functional DPs may move *in toto*.

However, the roadblock encountered in the attempt to incorporate Munn's proposal into Chierchia and Nissenbaum suggests that if a variable-based seman-

tics of functional extraction is correct, it must be the case that functional traces are indeed structured in the way that Cresti proposes. The interaction of functional extraction with parasitic gap constructions, then, sheds considerable light on the theory of functional extraction itself. That is, despite the fact that Cresti's proposal about the structure of functional traces involves syntactically unprecedented movement and hinges crucially on a tenuous argument that the individual-type argument of a functional trace is a null reflexive pronoun, the complication to the syntax and semantics of functional traces that she proposes is necessary for the composition of functional parasitic gaps in a standard variable-based theory.

4. Conclusion

Variable-free semantics provides an elegant, directly compositional analysis of functional extraction, and combinatory categorial grammar provides the same for parasitic gap constructions. Variable-based theories have some difficulty with both of these—I've argued here that the analysis of parasitic gaps requires the adoption of a non-compositional Parasitic Predicate Abstraction rule. In variable-free semantics, the interaction of the theory of functional extraction with that of parasitic gaps is seamless, and Munn's proposal may be straightforwardly implemented, accurately predicting that there are no two-binders readings for functional parasitic gaps. On the other hand, in a variable-based theory, Munn's proposal must be modified from a stipulation about types to one about indices, and even this modified proposal is not compatible with all variable-based theories of functional extraction. In fact, it is only compatible with Cresti's theory, in which functional traces are structured, bear at least three indices, and can move *in toto*. Cresti's is a relatively tenuous position that adds considerable complexity to the syntax and semantics of functional extraction. But the fact that single-binder functional parasitic gap constructions are grammatical forces variable-based theories of functional extraction to incorporate Cresti's analysis.

Endnotes

* Thank you to Polly Jacobson, Yael Sharvit, Jon Nissenbaum, and an anonymous reviewer for their extremely helpful comments.

[1] For an earlier proposal, see Gazdar, Klein, Pullum and Sag (1985).

[2] This formulation of Steedman's **S** is unarized to match Jacobson's formulation of **z**, but nothing hinges on this.

[3] It is hard to distinguish between the predictions made by a type *e* restriction and an NP restriction. But for the purposes of the present paper, I only explore the hypothesis that the restriction is, indeed, semantic.

[4] The abbreviation VP stands for S\NP.

[5] In the CCG I'm presenting here, gaps are definable from the point of view of the linguist, since a linguist can examine a derivation and find out where a verb or

preposition never got an argument. But the grammar doesn't recognize that "thinks that George fired" contains a gap any more than "fired" does—these two elements have the same syntactic category. So the term gap, as I use it here, is purely an informal descriptive term, not part of the theory.

[6] Nissenbaum actually proposes that these Spec of VP landing sites are *rightward* Specs to account for certain facts having to do with Heavy NP Shift, but this is not important for the discussion here.

[7] In fact, even the first clause of (19), which applies to assign an interpretation to VP0 in (20), is non-compositional, since the interpretation of VP0 depends on the interpretation of its sister, not just its daughters. But it is trivial to rewrite this clause of the rule in a compositional way (see (22), e.g.).

[8] There is guaranteed to be a unique highest-typed element, since either there is only a single type *e* variable, or one variable takes all the others as arguments.

[9] Thanks to an anonymous reviewer for pointing out that a functional extraction theory that builds LFs like (29) would solve the problem of incompatibility between theories.

References

Chierchia, Gennaro: 1993, 'Questions with Quantifiers', *Natural Language Semantics* **1**, 181–234.

Chomsky, Noam: 1986, *Barriers*, The MIT Press, Cambridge, Massachusetts.

Cresti, Diana: 1995, 'Extraction and Reconstruction', *Natural Language Semantics* **3**, 79–122.

Culicover, P. W., and P. M. Postal (eds): 2001, *Parasitic Gaps*, MIT Press, Cambridge, Massachusetts.

Curry, Haskell, and Robert Feys: 1958, *Combinatory Logic*, North Holland Publishing Company, Amsterdam.

Engdahl, Elizabet: 1986, *Constituent Questions*, D. Reidel Publishing Co., Dordrecht.

Gazdar, Gerald, Ewan Klein, Geoffrey Pullum, and Ivan Sag: 1985, *Generalized Phrase Structure Grammar*, Harvard University Press, Cambridge, Massachusetts.

Groenendijk, Jeroen, and Martin Stokhof: 1983, 'Interrogative Quantifiers and Skolem Functions', *in* K. Ehlich and H. van Riemsdijk (eds), *Connectedness in Sentence, Discourse, and Text*, Tilburg Studies in Language and Literature, Tilburg, pp. 71–110.

Heim, Irene, and Angelika Kratzer: 1998, *Semantics in Generative Grammar*, Blackwell Publishers, Oxford.

Jacobson, Pauline: 1999, 'Towards a Variable-Free Semantics', *Linguistics and Philosophy* **22**, 117–185.

Jacobson, Pauline: 2000, 'Paycheck Pronouns, Bach-Peters Sentences, and Variable-Free Semantics', *Natural Language Semantics* **8**, 77–155.

Jacobson, Pauline: 2002, 'The (Dis)organization of the Grammar: 25 years', *Lin-*

guistics and Philosophy **25**, 601–626.

Munn, Alan: 2001, 'Explaining Parasitic Gap Constructions', *in* P. W. Culicover and P. M. Postal (eds), *Parasitic Gaps*, MIT Press, Cambridge, Massachusetts.

Nissenbaum, Jon: 2000, *Investigations of Covert Phrase Movement*, PhD thesis, Massachusetts Institute of Technology.

Steedman, Mark: 1987, 'Combinatory Grammars and Parasitic Gaps', *Natural Language and Linguistic Theory* **5**, 403–439.

Szabolcsi, Anna, and Frans Zwarts: 1997, 'Weak Islands and an Algebraic Semantics for Scope Taking', *in* A. Szabolcsi (ed.), *Ways of Scope Taking*, Kluwer Academic Publishers, Dordrecht, pp. 217–262.