Grammar in Performance and Acquisition: interfaces

E Stabler, UCLA

ENS Paris • 2008 • day 2

Q1 How are utterances interpreted 'incrementally'?

Q2 How is that ability acquired, from available evidence?

Q3 Why are some constituent orders unattested across languages?

Q4 What kind of grammar makes copying a natural option?

- we don't need to start from zero (start from grammar)
- frame explanations supported by convergent evidence

- Everyone$_1$, someone saw t_1

  ‘Interpret t_1 as a variable x_1 bound by a higher abstraction.’
  everyone$(\lambda x_1.$**someone**$(\lambda x_2.\mathit{loves}\, x_1 x_2))$.

- Everyone$_1$, someone saw t_1

  ‘Interpret t_1 as a variable x_1 bound by a higher abstraction.’
  $\textbf{everyone}(\lambda x_1.\textbf{someone}(\lambda x_2.\mathit{loves}\, x_1 x_2)$.

PBC Each trace must be bound at S-structure.

GPBC Each trace must be bound throughout the derivation.

- Everyone$_1$, someone saw t_1

  'Interpret t_1 as a variable x_1 bound by a higher abstraction.'
  everyone$(\lambda x_1.$**someone**$(\lambda x_2.loves x_1 x_2)$.

PBC Each trace must be bound at S-structure.

GPBC Each trace must be bound throughout the derivation.

- [t_1 saw everyone]$_2$, someone$_1$ did t_2

PBC Each trace must be bound at S-structure.

GPBC Each trace must be bound throughout the derivation.

(Müller'98 and many others):

- $[_{VP_2}$ t_1 Gelesen] hat [das Buch$]_1$ [keiner t_2].
  read has the book noone
- $[_{VP_2}$Criticized by his boss t_1] John$_1$ has never been t_2.
- $[_{AP_2}$How likely [t_1 to win]] is$_3$ John$_1$ t_3 t_2?
- *$[_{AP_2}$How likely [t_1 to be a riot]] is$_3$ there$_1$ t_3 t_2?
- John $[_{VP_2}$reads t_1] [no novels$]_1$ t_2.

[T]he hypothesis of direct compositionality can be summed up with the following slogan:

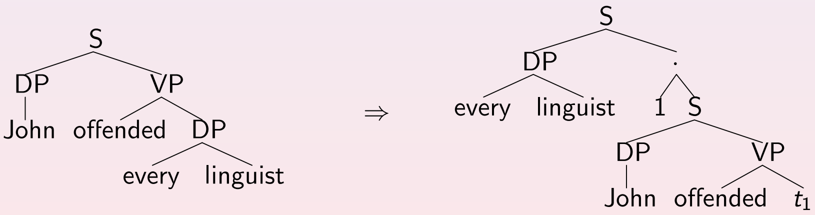
The syntax and the semantics work together in tandem.

. . . it ensures that . . . every expression which is computed in the syntax . . . actually does have a meaning. . .

To illustrate with a concrete example, consider the standard, non-directly compositional analysis of quantifier scope construal: a verb phrase such as *saw everyone* fails to have a semantic interpretation until it has been embedded within a large enough structure for the quantifier to take scope (e.g. *Someone saw everyone*). On such an analysis, there is no semantic value to assign to the verb phrase *saw everyone* at the point in the derivation in which it is first formed by the syntax (or any other point in the derivation, for that matter). (Barker and Jacobson, 2007, pp.1-2)

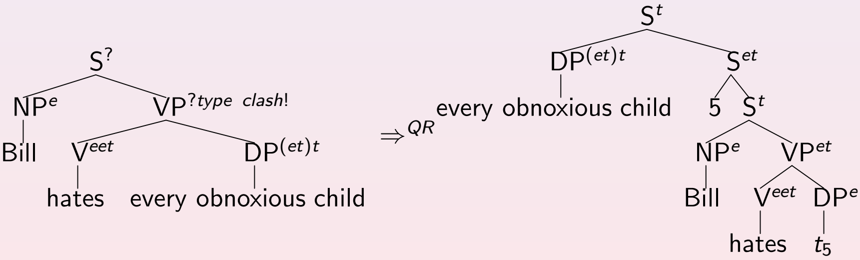
(let's worry about this simple case first!)

. . . the overt structure of “John offended every linguist” . . . cannot be the input to the semantic component. . . The DP “every linguist”. . . will move out of its VP and adjoin to S in the derivation from SS to LF.



(Heim&Kratzer’86,pp.184-5)

There is no way to assign a type to the VP-node in our system. . . The type clash is resolved by May's rule Quantifier Raising (QR). . .



(von Stechow'08)

Frege: quantifiers as properties of properties of individuals

It is true that at first sight the proposition

"All whales are mammals"

seems to be not about concepts but about animals; but if we ask which animal then we are speaking of, we are unable to point to any one in particular. . . If it be replied that what we are speaking of is not, indeed, an individual definite object, but nevertheless an indefinite object, I suspect that "indefinite object" is only another term for concept. . . (1884, §47)

. . . trouble with the semantic paradoxes ⇒ types

Syntax

Church'40 simple type theory (see e.g. Carpenter'97 text)

- Given a set of basic types $\mathbb{B}$, we build the whole set of **types**

$$\mathbb{T} := \mathbb{B} \mid (\mathbb{T}\mathbb{T}).$$

- $\forall \tau \in \mathbb{T}$, **vars** $\mathbf{V}^\tau$ $(x_0^\tau, x_1^\tau, \ldots)$ and **constants** $\mathbf{C}^\tau$ $(\mathbf{c}_0^\tau, \mathbf{d}_1^\tau, \ldots)$

$$\text{Terms } \Lambda := \mathbf{V}^\tau \mid \mathbf{C}^\tau \mid (\Lambda^{\sigma\tau}\Lambda^\sigma)^\tau \mid (\lambda \mathbf{V}^\sigma.\Lambda^\tau)^{\sigma\tau}$$

- **Notation:**

across types	$\mathbf{V} = \bigcup_\tau \mathbf{V}^\tau$		$\mathbf{C} = \bigcup_\tau \mathbf{C}^\tau$
types associate right	eet	$=$	$e(et)$
abstraction associates right	$\lambda x.\lambda y.\lambda z.M$	$=$	$\lambda x.(\lambda y.(\lambda z.M))$
applications associate left	$fabc$	$=$	$((fa)b)c)$
application over abstraction	$\lambda x.fxy$	$=$	$\lambda x.((fx)y)$

semantics

- For each basic type $\tau \in \mathbb{B}$, Dom_τ is a set
- For all other types $\mathbf{Dom}_{\alpha\beta} = [\mathbf{Dom}_\alpha \rightarrow \mathbf{Dom}_\beta]$
- Frame $\mathbf{Dom} = \bigcup_{\alpha \in \mathbf{Typ}} \mathbf{Dom}_\alpha$
- Model $\mathcal{M} = \langle \mathbf{Dom}, [\![.]\!] \rangle$, where

  Dom is a frame, and
  $[\![.]\!] : \mathbf{C} \rightarrow \mathbf{Dom}$ such that if $\alpha \in \mathbf{C}_\tau$ then $[\![\alpha]\!] \in \mathbf{Dom}_\tau$
- Assignments $\theta : V \rightarrow Dom$ such that $\theta(x) \in \mathbf{Dom}_\alpha$ if $x \in \mathbf{V}^\alpha$
- Denotations wrt $\mathcal{M}$ and θ,

  $[\![x]\!]^\theta_\mathcal{M} = \theta(x)$ if $x \in \mathbf{V}$,
  $[\![c]\!]^\theta_\mathcal{M} = [\![c]\!]$ if $c \in \mathbf{C}$,
  $[\![\alpha\beta]\!]^\theta_\mathcal{M} = [\![\alpha]\!]^\theta_\mathcal{M}[\![\beta]\!]^\theta_\mathcal{M}$,
  $[\![\lambda x.\alpha]\!]^\theta_\mathcal{M} = f$ such that $fa = [\![\alpha]\!]^{\theta[x:=a]}_\mathcal{M}$.

(Syn) Basic types $\mathbb{B} = \{e, t\}$, and for each type τ, constants

$$\begin{array}{ll} \mathbf{not} \in \mathbf{C}^{tt} & \mathbf{and} \in \mathbf{C}^{ttt} \\ \mathbf{eq}_\tau \in \mathbf{C}^{\tau\tau t} & \mathbf{everything}_\tau \in \mathbf{C}^{(\tau t)t} \\ \iota_\tau \in \mathbf{C}^{(\tau t)\tau} & \end{array}$$

(Sems) $\mathbf{Dom_t} = \{\text{true}, \text{false}\}$, $\mathbf{Dom_e}$ any set of individuals, and constants are interpreted as follows:

$[\![\mathbf{not}]\!](x) = \text{true if } x = \text{false, false otherwise}$

$[\![\mathbf{and}]\!](x)(y) = \text{true if } x = \text{true and } y = \text{true, false otherwise}$

$[\![\mathbf{eq}_\tau]\!](x)(y) = \text{true if } x = y\text{, false otherwise}$

$$[\![\mathbf{everything}_\tau]\!](P) = \begin{cases} \text{true} & \text{if } \forall \mathbf{a} \in \mathbf{Dom}_\tau, P(\mathbf{a}) = \text{true} \\ \text{false} & \text{otherwise} \end{cases}$$

$[\![\iota]\!](P) = a$ if a is the unique thing such that $P(a) = \text{true}$.

Instead of **everything**$^{(et)t}$, Church has Π and Carpenter has **every**, with **some**$^{(et)t}$ or **something** introduced by definition.

(limited) polymorphism

- Easy extensions are available for limited polymorphism.
  E.g. instead of $\mathbf{eq}_\tau$ for each type τ, in λ_2,

$$\mathbf{eq} = \Lambda\alpha.\lambda x^\alpha.x^\alpha$$

E.g. instead of type shifting (cf Capretta'02), $\forall n \in \mathbb{N}$

$$p_0 = t. \qquad p_{n+1} = ep_n. \qquad \mathbf{everything}^{p_{n+1}p_n}.$$

(Barendregt'92 survey linked on web page)

(Syn) Add logical constants **every**, **some** $\in \mathbf{C}^{(et)(et)t}$ and constants **person**, **thing** $\in \mathbf{C}^{et}$, **saw** $\in \mathbf{C}^{eet}$.

(Sem)

$$[\![\mathbf{every}\,PQ]\!] = \begin{cases} \text{true} & \text{if } Pa \rightarrow Qa, \text{ all } a \in \mathbf{Dom}_e \\ \text{false} & \text{otherwise} \end{cases}$$

$$[\![\mathbf{some}\,PQ]\!] = \begin{cases} \text{true} & \text{if } Pa = Qa = \textit{true}, \text{ some } a \in \mathbf{Dom}_e \\ \text{false} & \text{otherwise} \end{cases}$$

(E.g.) Then we have formulas like these

$$(\mathbf{every\ person})(\lambda y.(\mathbf{some\ thing})(\lambda x.\mathbf{saw}xy))$$

for VP=[binary relation+quantifier], two main approaches:

1. saturate relation, abstract to bind var, then apply quantifier (Heim&Kratzer, von Stechow,. . .)

$$(\textbf{some person})(\lambda y.(\textbf{every thing})(\lambda x.\textbf{saw}xy))$$

2. type-shift (Hendriks, Jacobson, Barker, Winter,. . .).

$$L = \lambda Q^{(et)t}.\lambda R^{eet}.\lambda y.Q(\lambda x.Rxy)$$
$$(\textbf{some person})(L(\textbf{every thing})\textbf{saw})$$

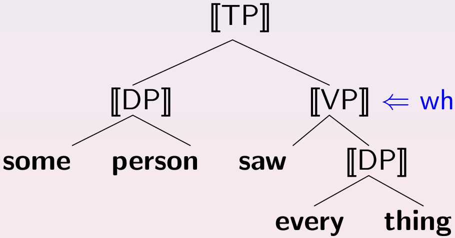
2'. simply assume quantifiers are polymorphic (Keenan,. . .)

$$(\textbf{some person})((\textbf{every thing})\textbf{saw})$$

(NB: in all 3 approaches, (**some person**) has the identical argument, provided by VP)

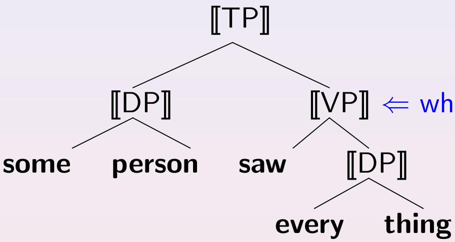
So let's adopt Keenan's simple 'arity reducer' perspective, but use QR to establish scope. . .

What could VP denotation be, on a standard QR story?



⇐ what could this be?

- Scope determined by ‘landing position’ of object.
- Roughly, from $[\![\mathrm{VP}]\!]$ we need (**every thing**)λx and **saw**x



$[\![\text{TP}]\!]$

$[\![\text{DP}]\!]$ $[\![\text{VP}]\!]$ $\Leftarrow$ what could this be?

some **person** **saw** $[\![\text{DP}]\!]$

every **thing**

- Roughly, from $[\![\text{VP}]\!]$ we need (**every thing**)λx and **saw**x

Two technical issues: (cf. Kobele'06, PL sems)

- Variable x has to be 'fresh' to avoid accidental capture
- What is λx?

But for MG interpretation, these issues can be avoided.

Basic idea:

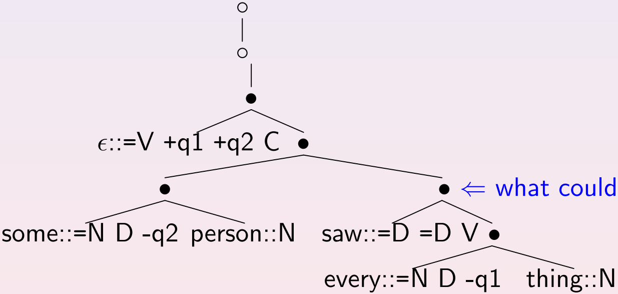
- In MGs, QR triggered by some feature of DP (e.g. -q,-top)
- Call a tree *useful* if it occurs in a completed derivation
- By SMC, no 2 constituents in any useful tree have the same initial licensee feature
- So if some subset of the licensee features $L = \{\text{-}f_1, \ldots, \text{-}f_k\}$, trigger DP movement to interpreted positions, we represent the meaning of each useful tree with a $k+1$-tuple:

  $$(s_0, s_1, \ldots, s_k),$$

  with s_0 the semantic value of the head, and each other s_i the value of the subtree (if any) moving for feature $\text{-}f_i$.
- We will consistently use variable x_i for feature s_i, so if a constituent moves first for $\text{-}f_1$ and then for $\text{-}f_2$, after the first movement we equate $x_1 = x_2$ and immediately bind x_1.

(similar association of variables with structural positions, with finite bounds, will be available with most modifications of the SMC considered on the first day)

Example 1a:



ϵ::=V +q1 +q2 C

some::=N D -q2 person::N

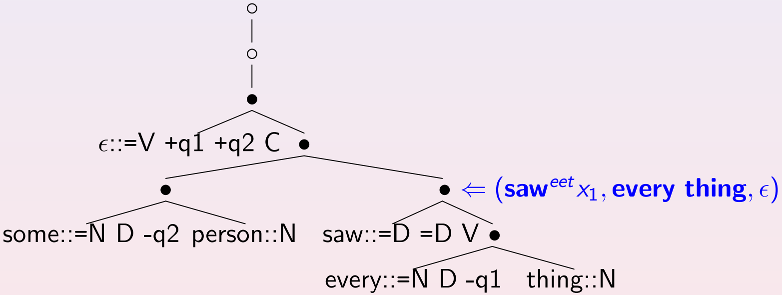
saw::=D =D V

every::=N D -q1 thing::N

⇐ what could this be?

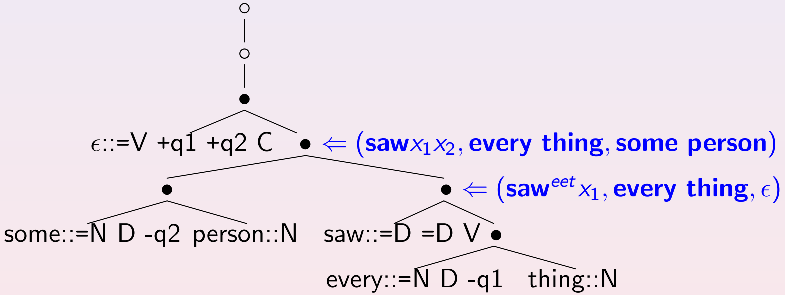
To make the VP easier to point to, I put subject first, but as usual the selected subj is the 2nd arg of *em*)

Example 1a:



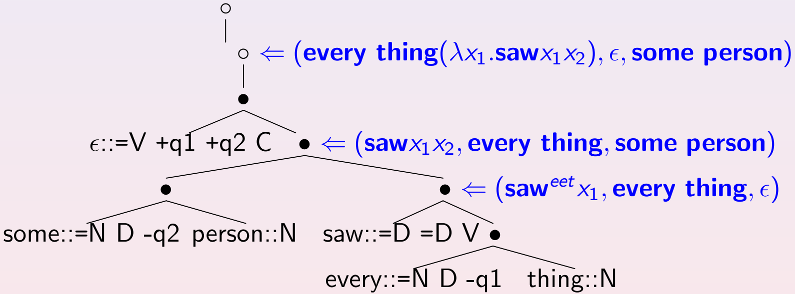
To make the VP easier to point to, I put subject first, but as usual the selected subj is the 2nd arg of *em*)

Example 1a:



To make the VP easier to point to, I put subject first, but as usual the selected subj is the 2nd arg of *em*)

Example 1a:



$\Leftarrow (\textbf{every thing}(\lambda x_1.\textbf{saw}\, x_1 x_2), \epsilon, \textbf{some person})$

$\Leftarrow (\textbf{saw}\, x_1 x_2, \textbf{every thing}, \textbf{some person})$

$\Leftarrow (\textbf{saw}^{eet} x_1, \textbf{every thing}, \epsilon)$

ϵ::=V +q1 +q2 C

some::=N D -q2

person::N

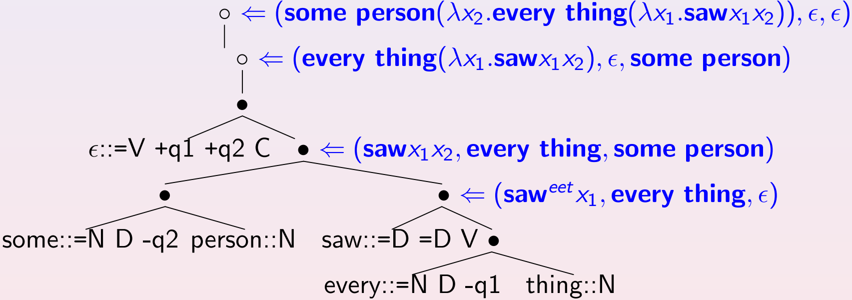
saw::=D =D V

every::=N D -q1

thing::N

To make the VP easier to point to, I put subject first, but as usual the selected subj is the 2nd arg of *em*)

Example 1a:



$\circ \Leftarrow (\textbf{some person}(\lambda x_2.\textbf{every thing}(\lambda x_1.\textbf{saw} x_1 x_2)), \epsilon, \epsilon)$

$\circ \Leftarrow (\textbf{every thing}(\lambda x_1.\textbf{saw} x_1 x_2), \epsilon, \textbf{some person})$

$\bullet$

ϵ::=V +q1 +q2 C

$\bullet \Leftarrow (\textbf{saw} x_1 x_2, \textbf{every thing}, \textbf{some person})$

$\bullet$

$\bullet \Leftarrow (\textbf{saw}^{eet} x_1, \textbf{every thing}, \epsilon)$

some::=N D -q2

person::N

saw::=D =D V

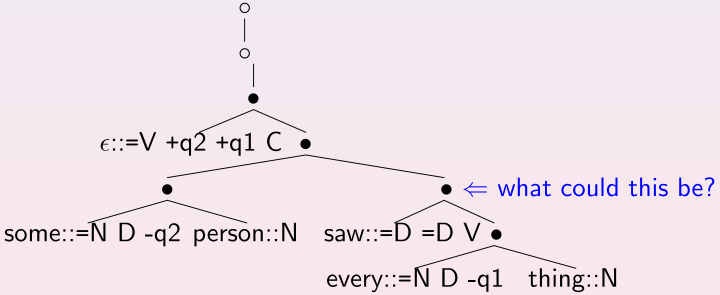
$\bullet$

every::=N D -q1

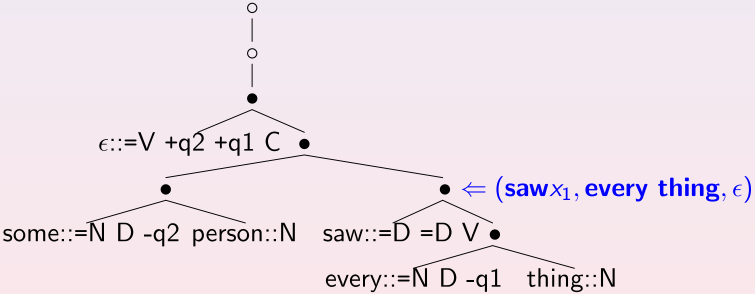
thing::N

To make the VP easier to point to, I put subject first, but as usual the selected subj is the 2nd arg of *em*)

Example 1b:


ϵ::=V +q2 +q1 C
some::=N D -q2
person::N
saw::=D =D V
every::=N D -q1
thing::N
⇐ what could this be?

Example 1b:



ϵ::=V +q2 +q1 C

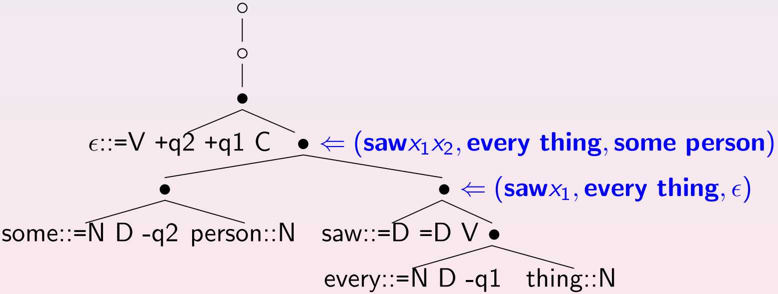
some::=N D -q2 person::N

saw::=D =D V

every::=N D -q1 thing::N

$\Leftarrow$ (**saw** x_1, **every thing**, ϵ)

Example 1b:



ϵ::=V +q2 +q1 C

$\Leftarrow$ (**saw**x_1x_2, **every thing**, **some person**)

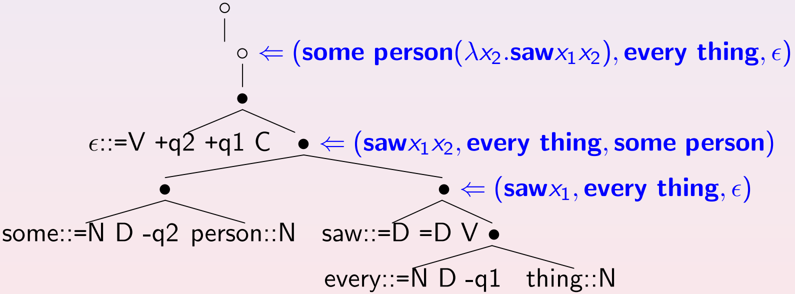
$\Leftarrow$ (**saw**x_1, **every thing**, ϵ)

some::=N D -q2 person::N

saw::=D =D V

every::=N D -q1 thing::N

Example 1b:



○ ⇐ $(\textbf{some person}(\lambda x_2.\textbf{saw}\, x_1 x_2), \textbf{every thing}, \epsilon)$

ϵ::=V +q2 +q1 C

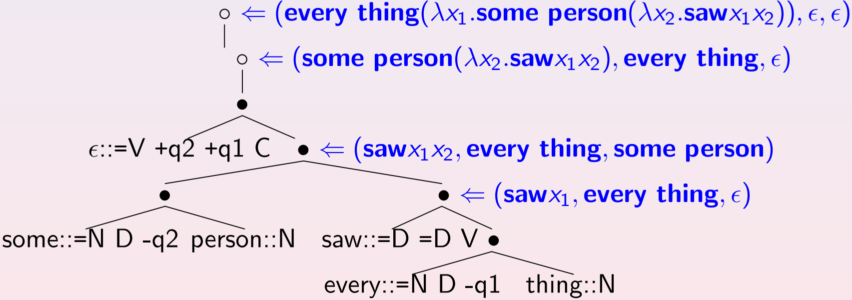
● ⇐ $(\textbf{saw}\, x_1 x_2, \textbf{every thing}, \textbf{some person})$

● ⇐ $(\textbf{saw}\, x_1, \textbf{every thing}, \epsilon)$

some::=N D -q2 person::N saw::=D =D V

every::=N D -q1 thing::N

Example 1b:



○ ⇐ $(\textbf{every thing}(\lambda x_1.\textbf{some person}(\lambda x_2.\textbf{saw}x_1x_2)), \epsilon, \epsilon)$

○ ⇐ $(\textbf{some person}(\lambda x_2.\textbf{saw}x_1x_2), \textbf{every thing}, \epsilon)$

●

ϵ::=V +q2 +q1 C

● ⇐ $(\textbf{saw}x_1x_2, \textbf{every thing}, \textbf{some person})$

●

● ⇐ $(\textbf{saw}x_1, \textbf{every thing}, \epsilon)$

some::=N D -q2

person::N

saw::=D =D V

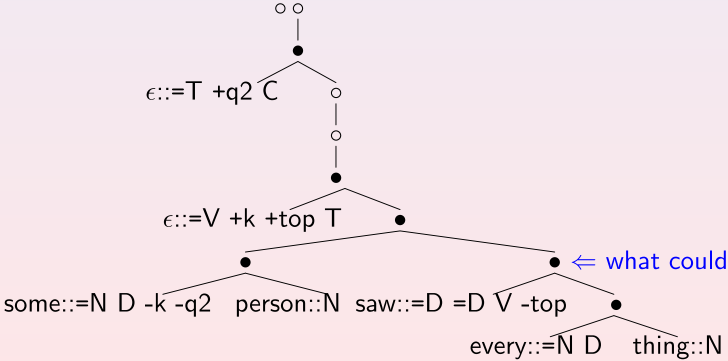
●

every::=N D -q1

thing::N

Hiraiwa'02: Someone saw everyone $(\exists > \forall, \forall > \exists)$
Saw everyone, someone did $(\exists > \forall, *\forall > \exists)$

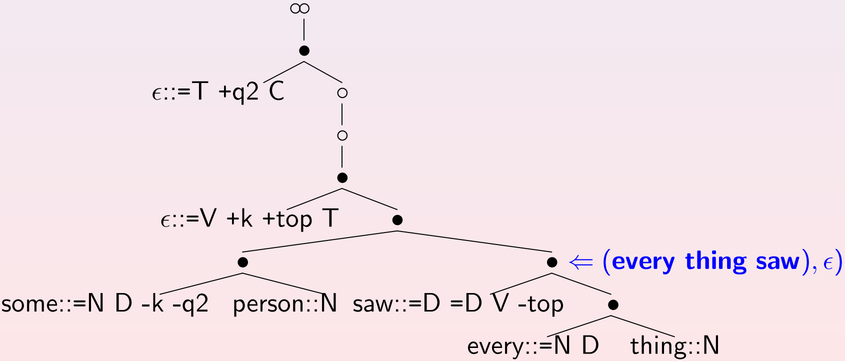
$*\forall > \exists$ if SpIC or other constraint blocks q-movement from spec,TP



⇐ what could this be?

Hiraiwa'02: Someone saw everyone $(\exists > \forall, \forall > \exists)$
Saw everyone, someone did $(\exists > \forall, *\forall > \exists)$

$*\forall > \exists$ if SpIC or other constraint blocks q-movement from spec,TP



∞

ϵ::=T +q2 C

ϵ::=V +k +top T

⇐ (every thing saw), ϵ)

some::=N D -k -q2

person::N

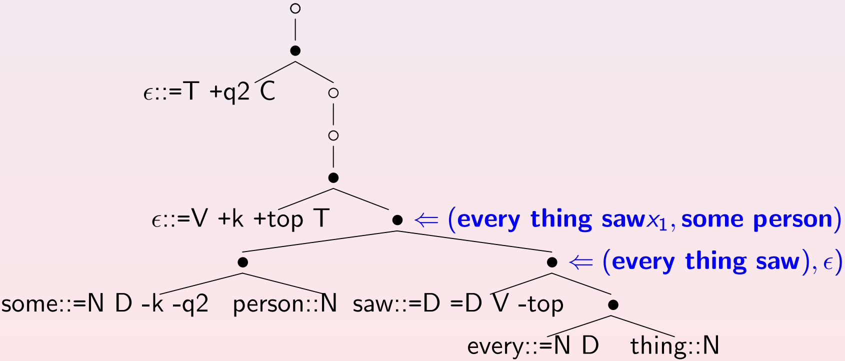
saw::=D =D V -top

every::=N D

thing::N

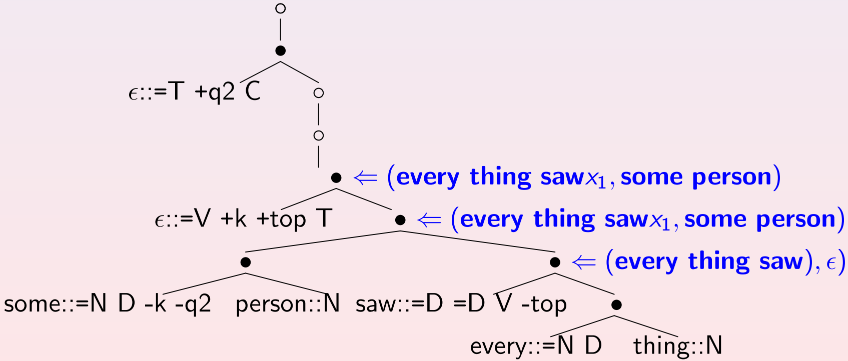
Hiraiwa'02: Someone saw everyone $(\exists > \forall, \forall > \exists)$
Saw everyone, someone did $(\exists > \forall, *\forall > \exists)$

$*\forall > \exists$ if SpIC or other constraint blocks q-movement from spec,TP



Hiraiwa'02: Someone saw everyone $(\exists > \forall, \forall > \exists)$
Saw everyone, someone did $(\exists > \forall, *\forall > \exists)$

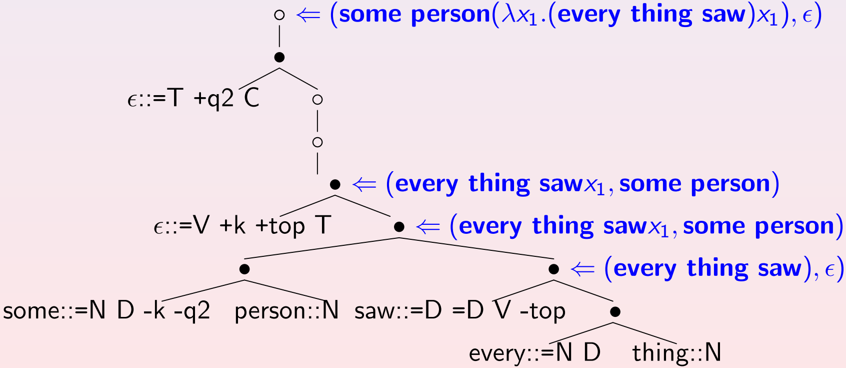
$*\forall > \exists$ if SpIC or other constraint blocks q-movement from spec,TP



○
●
ϵ::=T +q2 C
○
○
● $\Leftarrow$ (**every thing saw**x_1, **some person**)
ϵ::=V +k +top T
● $\Leftarrow$ (**every thing saw**x_1, **some person**)
●
● $\Leftarrow$ (**every thing saw**), ϵ)
some::=N D -k -q2
person::N
saw::=D =D V -top
●
every::=N D
thing::N

Hiraiwa'02: Someone saw everyone $(\exists > \forall, \forall > \exists)$
Saw everyone, someone did $(\exists > \forall, *\forall > \exists)$

$*\forall > \exists$ if SpIC or other constraint blocks q-movement from spec,TP



$\circ \Leftarrow$ (**some person**$(\lambda x_1.$(**every thing saw**)$x_1), \epsilon)$

ϵ::=T +q2 C

$\bullet \Leftarrow$ (**every thing saw**x_1, **some person**)

ϵ::=V +k +top T

$\bullet \Leftarrow$ (**every thing saw**x_1, **some person**)

$\bullet \Leftarrow$ (**every thing saw**), ϵ)

some::=N D -k -q2 person::N saw::=D =D V -top

every::=N D thing::N

- Consider MG with some subset of features $L = \{\text{-}f_1, \ldots, \text{-}f_k\}$ (including e.g. -q, -foc), triggering DP movement to clause peripheral positions where they can be interpreted. Everything else interpreted in base position.
- Tree t is *useful* iff it occurs in a completed derivation
- Interpret useful tree t as a tuple, $[\![t]\!] = (s_0, s_1, \ldots, s_k)$ where
    - s_0 is the semantic value of the t-head, and for $1 \leq i \leq k$,
    - s_i is the semantic value of the $-f_i$ head, if any, otherwise ϵ
- Given $(s_0, \ldots, s_k)^{[i:=x]} = (s_0, \ldots, s_{i-1}, x, s_{i+1}, \ldots s_k)$

  (Sometimes we have a sequence of substitutions to make $[i_1 := x_1, \ldots, i_n := x_n]$, all i_j distinct)
- Given $s = (s_0, s_1, \ldots, s_k)$ and $t = (t_0, t_1, \ldots, t_k)$ let $(s{+}t) = (u_0, \ldots, u_k)$ where $u_i = s_i$ if $s_i \neq \epsilon$, else $u_i = t_i$.
- $FF(t) = f$ means the first feature of head of tree t is f

$$\begin{array}{llll} \text{For} & t_1[\text{=}c] = a & \text{with} & [\![a]\!] = (s_0, \ldots, s_k), \\ \text{and} & t_2[c] = b & \text{with} & [\![b]\!] = (r_0, \ldots, r_k), \end{array}$$

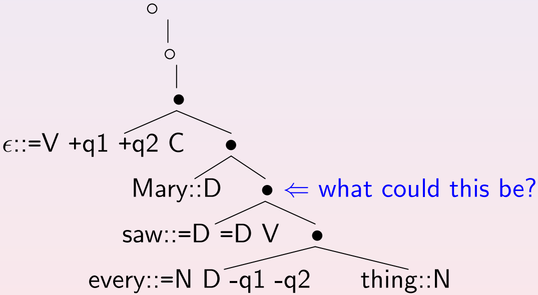
$$[\![\mathbf{em}(a,b)]\!] = \begin{cases} ([\![a]\!]+[\![b]\!])^{[0:=s_0 x_i,\, i:=r_0]} & \text{if } FF(t_2) = \text{-}f_i \in L \quad (store) \\ ([\![a]\!]+[\![b]\!])^{[0:=s_0 r_0]} & \text{if } s_0 r_0 \text{ well-typed} \quad (FA) \\ ([\![a]\!]+[\![b]\!])^{[0:=r_0 s_0]} & \text{otherwise} \quad (BA) \end{cases}$$

$$\text{For} \quad t_1[\text{+}f_j] = a \quad \text{with} \quad [\![a]\!] = (s_0, \ldots, s_k), \text{ with subtree } t_2[-f_j],$$

$$[\![\mathbf{im}(a)]\!] = \begin{cases} [\![a]\!] & \text{if } FF(t_2) = \text{-}f_i,\, i = j \quad (ck) \\ [\![a]\!]^{[0:=some(\lambda x_j . x_i = x_j \wedge s_0),\ i:=s_j,\ j:=\epsilon]} & \text{if } FF(t_2) = \text{-}f_i \in L \quad (ck) \\ [\![a]\!] & \text{if } FF(t_2) \notin L \quad (0) \\ [\![a]\!]^{[j:=\epsilon,\ 0:=s_j(\lambda x_j . s_0)]} & \text{if } t_2 \text{ has no features.} \quad (bnd) \end{cases}$$

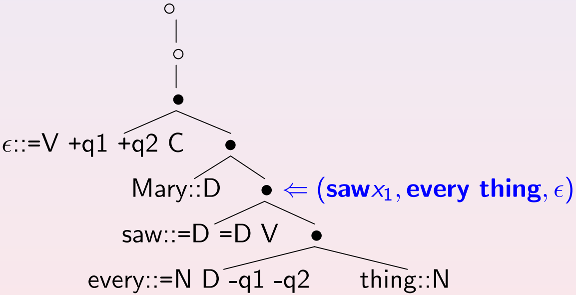
(nb: in these defs, sequences of cases are to be understood in order, as if…else, and **some** is obviously $(et)t$.)

checking example:



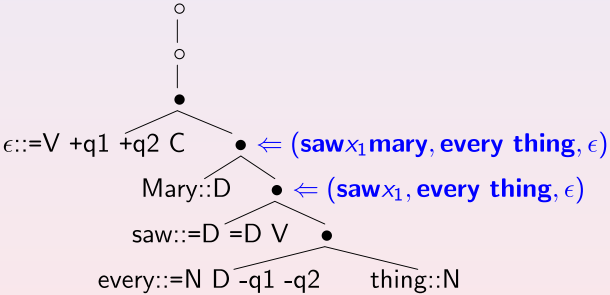
Again, to make the VP easier to point to, I put subject first, but selected subj is the 2nd arg of *em*)

checking example:



Again, to make the VP easier to point to, I put subject first, but selected subj is the 2nd arg of *em*)

checking example:



○

○

●

ϵ::=V +q1 +q2 C

● $\Leftarrow$ (**saw**x_1**mary**, **every thing**, ϵ)

Mary::D

● $\Leftarrow$ (**saw**x_1, **every thing**, ϵ)

saw::=D =D V

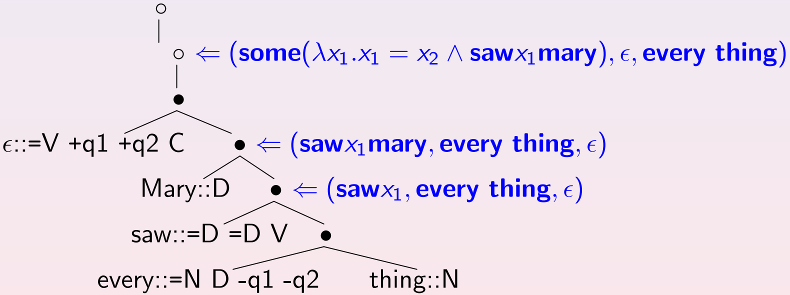
●

every::=N D -q1 -q2

thing::N

Again, to make the VP easier to point to, I put subject first, but selected subj is the 2nd arg of *em*)

checking example:



$\circ$

$\circ \Leftarrow (\textbf{some}(\lambda x_1 . x_1 = x_2 \wedge \textbf{saw}\,x_1\,\textbf{mary}), \epsilon, \textbf{every thing})$

$\bullet$

ϵ::=V +q1 +q2 C

$\bullet \Leftarrow (\textbf{saw}\,x_1\,\textbf{mary}, \textbf{every thing}, \epsilon)$

Mary::D

$\bullet \Leftarrow (\textbf{saw}\,x_1, \textbf{every thing}, \epsilon)$

saw::=D =D V

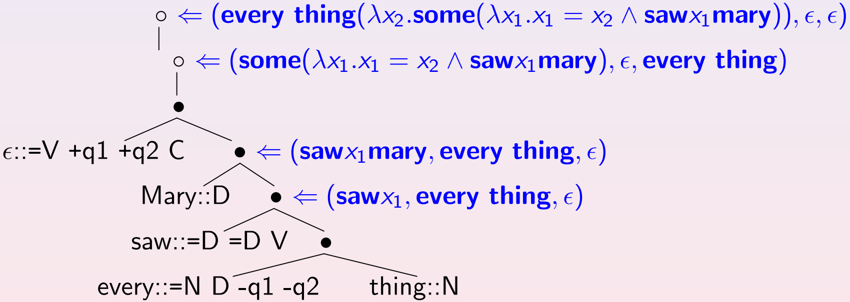
$\bullet$

every::=N D -q1 -q2

thing::N

Again, to make the VP easier to point to, I put subject first, but selected subj is the 2nd arg of *em*)

checking example:



$\circ \Leftarrow (\textbf{every thing}(\lambda x_2.\textbf{some}(\lambda x_1.x_1 = x_2 \wedge \textbf{saw}\,x_1\,\textbf{mary})), \epsilon, \epsilon)$

$\circ \Leftarrow (\textbf{some}(\lambda x_1.x_1 = x_2 \wedge \textbf{saw}\,x_1\,\textbf{mary}), \epsilon, \textbf{every thing})$

$\bullet$

ϵ::=V +q1 +q2 C

$\bullet \Leftarrow (\textbf{saw}\,x_1\,\textbf{mary}, \textbf{every thing}, \epsilon)$

Mary::D

$\bullet \Leftarrow (\textbf{saw}\,x_1, \textbf{every thing}, \epsilon)$

saw::=D =D V

$\bullet$

every::=N D -q1 -q2

thing::N

Again, to make the VP easier to point to, I put subject first, but selected subj is the 2nd arg of *em*)

Improvements

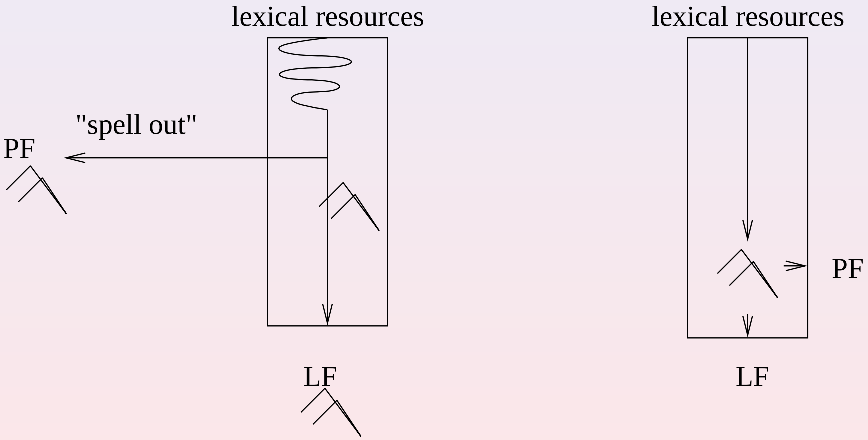
- can we make $[\![\text{move}]\!]$ uniform?
- when we don't know scope of object, does anything more follow about VP denotation that the pairs do not make explicit?

Improvements

- can we make ⟦move⟧ uniform?
  yes, and this gets us closer the the representational perspective – but too much for today
- when we don't know scope of object, does anything more follow about VP denotation that the pairs do not make explicit?

Improvements

- can we make $[\![\text{move}]\!]$ uniform?
  yes, and this gets us closer the the representational perspective – but too much for today
- when we don't know scope of object, does anything more follow about VP denotation that the pairs do not make explicit?
  yes, remember conservativity! This is important and usually ignored (but cf. Ben-Shalom, Keenan) – too much for today


lexical resources
"spell out"
PF
LF
lexical resources
PF
LF

Morphophonology as transduction

the king eat -s the pie ↦ the king eats the pie

(CL: Roark&Sproat'07,Huet'03)
(Ph: Riggle'04,Eisner'97)

Morphophonology as transduction

the king eat -s the pie	↦	the king eats the pie
the king have -s eat -en the pie	↦	the king has eaten the pie

(CL: Roark&Sproat'07,Huet'03)
(Ph: Riggle'04,Eisner'97)

Morphophonology as transduction

the king eat -s the pie ↦ the king eats the pie

the king have -s eat -en the pie ↦ the king has eaten the pie

the king have -s laugh -en ↦ the king has laughed

(CL: Roark&Sproat'07,Huet'03)

(Ph: Riggle'04,Eisner'97)

Morphophonology as transduction

the king eat -s the pie	↦	the king eats the pie
the king have -s eat -en the pie	↦	the king has eaten the pie
the king have -s laugh -en	↦	the king has laughed
the king be -s laugh -ing	↦	the king's laughing

(CL: Roark&Sproat'07,Huet'03)
(Ph: Riggle'04,Eisner'97)

Morphophonology as transduction

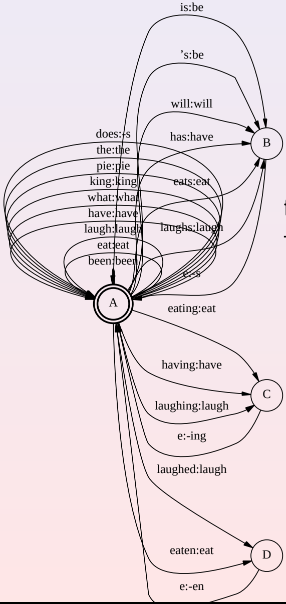
the king eat -s the pie	↦	the king eats the pie
the king have -s eat -en the pie	↦	the king has eaten the pie
the king have -s laugh -en	↦	the king has laughed
the king be -s laugh -ing	↦	the king's laughing
the king will -s laugh	↦	the king will laugh

(CL: Roark&Sproat'07,Huet'03)
(Ph: Riggle'04,Eisner'97)

Morphophonology as transduction

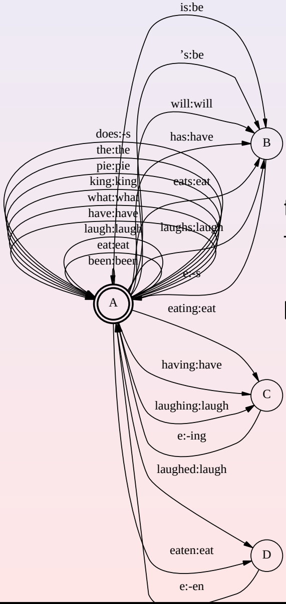
the king eat -s the pie	↦	the king eats the pie
the king have -s eat -en the pie	↦	the king has eaten the pie
the king have -s laugh -en	↦	the king has laughed
the king be -s laugh -ing	↦	the king's laughing
the king will -s laugh	↦	the king will laugh
-s the king laugh	↦	does the king laugh

(CL: Roark&Sproat'07,Huet'03)
(Ph: Riggle'04,Eisner'97)



forward:

the king eat -s the pie ↦ the king eats the pie



forward:

the king eat -s the pie ↦ the king eats the pie

backward:

the king eats the pie ↦ the king eat -s the pie
the king eat does the pie

- simple formalisms can model many linguistic proposals
- a straightforward semantics values every constituent in course of derivation
  - Simple, extensional MG semantics is defined in ≈7 lines
  - No problem with remnant movement
  - Conditions could be placed on use of LF variables (cf Collins&Sabel)
  - A tighter connection than pairing for VP is possible
- PF standardly handled by transducer composition

Q1 What performance models allow incremental interpretation (and remnant movement, doubling constructions)?

MG semantics

$$\begin{array}{llll} \text{For} & t_1[\text{=}c] = a & \text{with} & \llbracket a \rrbracket = (s_0, \ldots, s_k), \\ \text{and} & t_2[c] = b & \text{with} & \llbracket b \rrbracket = (r_0, \ldots, r_k), \end{array}$$

$$\llbracket \mathbf{em}(a,b) \rrbracket = \begin{cases} (\llbracket a \rrbracket + \llbracket b \rrbracket)^{[0:=s_0 x_i,\, i:=r_0]} & \text{if } FF(t_2) = \text{-}f_i \in L \quad (store) \\ (\llbracket a \rrbracket + \llbracket b \rrbracket)^{[0:=s_0 r_0]} & \text{if } s_0 r_0 \text{ well-typed} \quad (FA) \\ (\llbracket a \rrbracket + \llbracket b \rrbracket)^{[0:=r_0 s_0]} & \text{otherwise} \quad (BA) \end{cases}$$

For $t_1[+f_j] = a$ with $\llbracket a \rrbracket = (s_0, \ldots, s_k)$, with subtree $t_2[-f_j]$,

$$\llbracket \mathbf{im}(a) \rrbracket = \begin{cases} \llbracket a \rrbracket & \text{if } FF(t_2) = \text{-}f_i,\, i = j \quad (ck) \\ \llbracket a \rrbracket^{[0:=some(\lambda x_j . x_i = x_j \wedge s_0),\ i:=s_j,\ j:=\epsilon]} & \text{if } FF(t_2) = \text{-}f_i \in L \quad (ck) \\ \llbracket a \rrbracket & \text{if } FF(t_2) \notin L \quad (0) \\ \llbracket a \rrbracket^{[j:=\epsilon,\ 0:=s_j(\lambda x_j . s_0)]} & \text{if } t_2 \text{ has no features.} \quad (bnd) \end{cases}$$

Barendregt, Henk. 1992. Lambda calculi with types. In S. Abramsky, D.M. Gabbay, and T.S.E. Maibaum, editors, *Handbook of Logic in Computer Science, Volume 2*. Oxford University Press, Oxford.

Barker, Chris and Pauline Jacobson. 2007. Introduction: Direct compositionality. In Chris Barker and Pauline Jacobson, editors, *Direct Compositionality*. Oxford University Press, NY.

Barwise, Jon and Robin Cooper. 1981. Generalized quantifiers and natural language. *Linguistics and Philosophy*, 4:159–219.

Ben Shalom, Dorit. 1993. Object wide scope and semantic trees. In *Procs. of the Workshop on Semantics and Linguistic Theory, SALT III*.

Ben Shalom, Dorit. 1994. A tree characterization of generalized quantifier reducibility. In Makoto Kanazawa and Christopher J. Pi nón, editors, *Dynamics, Polarity and Quantification*. CSLI Publications, Stanford, California.

Ben Shalom, Dorit. 1996. *Semantic Trees*. Ph.D. thesis, University of California, Los Angeles.

Capretta, Venanzio. 2002. *Abstraction and Computation: Type Theory, Algebraic Structures, and Recursive Functions*. Ph.D. thesis, Katholieke Universiteit Nijmegen.

Carpenter, Bob. 1997. *Type-Logical Semantics*. MIT Press, Cambridge, Massachusetts.

Church, Alonzo. 1940. A formulation of the simple theory of types. *Journal of Symbolic Logic*, 5:56–68.

Cooper, Robin. 1983. *Quantification and Syntactic Theory*. Reidel, Dordrecht.

Eisner, Jason. 1997a. Efficient generation in Primitive Optimality Theory. In *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics*.

Eisner, Jason. 1997b. What constraints should OT allow? Presented at the Annual Meeting of the Linguistic Society of America, Chicago, January.

Frege, Gottlob. 1884. *Die Grundlagen der Arithmetik*. Koebner, Breslau. J.L. Austin's translation available as *The Foundations of Arithmetic*, Evanston, Illinois: Northwestern University Press, 1980.

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

Hendriks, Herman. 1993. *Studied Flexibility: Categories and types in syntax and semantics*. Ph.D. thesis, Universiteit van Amsterdam.

Henkin, Leon. 1950. Completeness in the theory of types. *Journal of Symbolic Logic*, 15:81–91.

Hiraiwa, Ken. 2002. Movement and derivation: Eliminating the PBC. In *Penn Linguistics Colloquium 26*.

Huet, Gérard P. 2003. Zen and the art of symbolic computing: Light and fast applicative algorithms for computational linguistics. In Verónica Dahl and Philip Wadler, editors, *Practical Aspects of Declarative Languages, 5th International Symposium, PADL 2003*, volume 2562 of *Lecture Notes in Computer Science*. Springer.

Keenan, Edward L. 1996. Further beyond the Frege boundary. In J. van der Does and J. van Eijck, editors, *Quantifiers, Logic, and Language*. CSLI Publications, Amsterdam.

Keenan, Edward L. and Dag Westerståhl. 1996. Generalized quantifiers in linguistics and logic. In Johan van Benthem and Alice ter Meulen, editors, *Handbook of Logic and Linguistics*. Kluwer, Boston.

Kobele, Gregory M. and Jens Michaelis. 2005. Two type 0 variants of minimalist grammars. In *Proceedings of the 10th conference on Formal Grammar and the 9th Meeting on Mathematics of Language, FGMOL05*.

May, Robert. 1977. *The Grammar of Quantification*. Ph.D. thesis, Massachusetts Institute of Technology.

May, Robert. 1985. *Logical Form: Its Structure and Derivation*. MIT Press, Cambridge, Massachusetts.

Michaelis, Jens. 2002. Note on the complexity of complex heads in a minimalist grammar. In *Proceedings of the 6th International Workshop on Tree Adjoining Grammars and Related Frameworks, TAG+6*.

Michaelis, Jens, Uwe Mönnich, and Frank Morawietz. 2000. Algebraic description of derivational minimalism. In *International Conference on Algebraic Methods in Language Proceeing, AMiLP'2000/TWLT16*.

Riggle, Jason. 2004. *Generation, Recognition, and Learning in Finite State Optimality Theory*. Ph.D. thesis, University of California, Los Angeles.

Roark, Brian and Richard Sproat. 2007. *Computational Approaches to Morphology and Syntax*. Oxford, NY.

Seki, Hiroyuki, Takashi Matsumura, Mamoru Fujii, and Tadao Kasami. 1991. On multiple context-free grammars. *Theoretical Computer Science*, 88:191–229.

Shieber, Stuart and Mark Johnson. 1994. Variations on incremental interpretation. *Journal of Psycholinguistic Research*, 22:287–318.

Stabler, Edward P. 1991. Avoid the pedestrian's paradox. In Robert C. Berwick, Steven P. Abney, and Carol Tenny, editors, *Principle-based Parsing: Computation and Psycholinguistics*. Kluwer, Boston, pages 199–238.

Steedman, Mark J. 1989. Grammar, interpretation, and processing from the lexicon. In William Marslen-Wilson, editor, *Lexical Representation and Process*. MIT Press, Cambridge, Massachusetts, pages 463–504.

van Benthem, Johan. 1984. Questions about quantifiers. *Journal of Symbolic Logic*, 49:443–466.

von Stechow, Arnim. 2008. Syntax and semantics: An overview. Ms., Universität Tübingen.

Westerståhl, Dag. 1994. Some results on quantifiers. *Notre Dame Journal of Formal Logic*, 25(2):152–170.

Winter, Yoad. 2003. Type shifting with semantic features: A unified perspective. In C. Barker and P. Jacobson, editors, *Direct Compositionality*. Oxford University Press, Oxford.