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
Everyone₁, someone saw $t₁$

‘Interpret $t₁$ as a variable $x₁$ bound by a higher abstraction.’

$\text{everyone}(\lambda x₁.\text{someone}(\lambda x₂.\text{loves}x₁x₂)$.
Everyone₁, someone saw $t_1$

‘Interpret $t_1$ as a variable $x_1$ bound by a higher abstraction.’

$$\text{everyone}(\lambda x_1. \text{someone}(\lambda x_2.\text{loves}x_1x_2)).$$

**PBC** Each trace must be bound at S-structure.

**GPBC** Each trace must be bound throughout the derivation.
Everyone₁, someone saw $t_1$

‘Interpret $t_1$ as a variable $x_1$ bound by a higher abstraction.’

$\text{everyone}(\lambda x_1.\text{someone}(\lambda x_2.\text{loves}x_1x_2))$.

**PBC** Each trace must be bound at S-structure.

**GPBC** Each trace must be bound throughout the derivation.

[t₁ saw everyone]₂, someone₁ 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 none

- $[VP_2 \ Criticized \ by \ his \ boss \ t_1] \ John_1 \ has \ never \ been \ t_2$.

- $[AP_2 \ How \ likely \ [t_1 \ to \ win]] \ is_3 \ John_1 \ t_3 \ t_2$?

- $*[AP_2 \ How \ likely \ [t_1 \ to \ be \ a \ riot]] \ is_3 \ there_1 \ t_3 \ t_2$?

- 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
Quantifiers and Arguments

Simple type theory (Church’40)
Extensions
Quantifiers

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:

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

- $\forall \tau \in T$, \textbf{vars} $V^\tau (x_0^\tau, x_1^\tau, \ldots)$ and \textbf{constants} $C^\tau (c_0^\tau, d_1^\tau, \ldots)$

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

- \textbf{Notation:}

  - across types $V = \bigcup_{\tau} V^\tau$ $C = \bigcup_{\tau} C^\tau$

  - 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 $\lambda x.fxy = \lambda x.((fx)y)$
quantifiers and arguments

Simple type theory (Church’40)
Extensions
Quantifiers

semantics

For each basic type $\tau \in \mathbb{B}$, $\text{Dom}_\tau$ is a set
For all other types $\text{Dom}_{\alpha\beta} = [\text{Dom}_\alpha \to \text{Dom}_\beta]$
Frame $\text{Dom} = \bigcup_{\alpha \in \text{Typ}} \text{Dom}_\alpha$
Model $\mathcal{M} = \langle \text{Dom}, \llbracket . \rrbracket \rangle$, where
- $\text{Dom}$ is a frame, and
- $\llbracket . \rrbracket : \mathcal{C} \to \text{Dom}$ such that if $\alpha \in \mathcal{C}_\tau$ then $\llbracket \alpha \rrbracket \in \text{Dom}_\tau$
Assignments $\theta : V \to \text{Dom}$ such that $\theta(x) \in \text{Dom}_\alpha$ if $x \in V^\alpha$
Denotations wrt $\mathcal{M}$ and $\theta$,
$$\llbracket x \rrbracket^\theta_{\mathcal{M}} = \theta(x) \text{ if } x \in V,$$
$$\llbracket c \rrbracket^\theta_{\mathcal{M}} = \llbracket c \rrbracket \text{ if } c \in \mathcal{C},$$
$$\llbracket \alpha\beta \rrbracket^\theta_{\mathcal{M}} = \llbracket \alpha \rrbracket^\theta_{\mathcal{M}} \llbracket \beta \rrbracket^\theta_{\mathcal{M}},$$
$$\llbracket \lambda x.\alpha \rrbracket^\theta_{\mathcal{M}} = f \text{ such that } fa = \llbracket \alpha \rrbracket^\theta_{\mathcal{M}}[x:=a].$$
(Syn) Basic types $\mathbb{B} = \{e, t\}$, and for each type $\tau$, constants

$$\begin{align*}
\text{not} & \in C^{tt} \\
\text{eq}_\tau & \in C^{\tau\tau t} \\
\iota_\tau & \in C^{(\tau t)\tau}
\end{align*}$$

and $\in C^{ttt}$

(Sems) $\text{Dom}_t = \{\text{true}, \text{false}\}$, $\text{Dom}_e$ any set of individuals, and constants are interpreted as follows:

$$\begin{align*}
[\text{not}] (x) & = \text{true} \text{ if } x = \text{false}, \text{ false otherwise} \\
[\text{and}] (x)(y) & = \text{true} \text{ if } x = \text{true} \text{ and } y = \text{true}, \text{ false otherwise} \\
[\text{eq}_\tau] (x)(y) & = \text{true} \text{ if } x = y, \text{ false otherwise} \\
[\text{everything}_\tau] (P) & = \begin{cases} 
\text{true} & \text{if } \forall a \in \text{Dom}_\tau, P(a) = \text{true} \\
\text{false} & \text{otherwise}
\end{cases} \\
[\iota] (P) & = a \text{ if } a \text{ is the unique thing such that } P(a) = \text{true}.
\end{align*}$$

Instead of $\text{everything}^{(et)t}$, Church has $\Pi$ and Carpenter has every, with $\text{some}^{(et)t}$ or something introduced by definition.
Easy extensions are available for limited polymorphism. E.g. instead of $\text{eq}_\tau$ for each type $\tau$, in $\lambda_2$, 

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

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

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

(Barendregt’92 survey linked on web page)
(Syn) Add logical constants every, some ∈ \( C^{et}(et)t \)
and constants person, thing ∈ \( C^{et} \), saw ∈ \( C^{eet} \).

(Sem) 
\[
\begin{align*}
\llbracket \text{every}PQ \rrbracket &= \begin{cases} 
true & \text{if } Pa \rightarrow Qa, \text{ all } a \in \text{Dom}_e \\
false & \text{otherwise}
\end{cases} \\
\llbracket \text{some}PQ \rrbracket &= \begin{cases} 
true & \text{if } Pa = Qa = true, \text{ some } a \in \text{Dom}_e \\
false & \text{otherwise}
\end{cases}
\end{align*}
\]

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

\[
(\text{every person})(\lambda y.(\text{some thing})(\lambda x.\text{saw}xy))
\]
for VP=[binary relation+quantifier], two main approaches:

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

\[(\text{somes person})(\lambda y.(\text{every thing})(\lambda x.\text{saw}xy))\]

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

\[L = \lambda Q^{(et)}t.\lambda R^{eet}.\lambda y.Q(\lambda x.Rxy)\]
\[(\text{somes person})(L(\text{every thing})\text{saw})\]

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

\[(\text{somes person})(((\text{every thing})\text{saw})\text{saw})\]

(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?

\[
\begin{array}{c}
\text{[TP]} \\
\text{[DP]} & \text{[VP]} & \iff \text{what could this be?} \\
\text{some} & \text{person} & \text{saw} \\
\text{every} & \text{thing} & \\
\end{array}
\]

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

Two technical issues: (cf. Kobele’06, PL sens)

- Variable $x$ has to be ‘fresh’ to avoid accidental capture
- What is $\lambda x$?

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 \( L = \{ -f_1, \ldots, -f_k \} \), 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 \(-f_i\).

- We will consistently use variable \( x_i \) for feature \( s_i \), so if a constituent moves first for \(-f_1\) and then for \(-f_2\), 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:

\[
\epsilon ::= V + q_1 + q_2 C
\]

\[
some ::= N D - q_2 \quad \text{person} ::= N
\]

\[
saw ::= D = D V
\]

\[
every ::= N D - q_1 \quad \text{thing} ::= N
\]

\[
\Rightarrow \quad \text{what could this be?}
\]

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:

\[ \epsilon ::= \text{V} + q1 + q2 \text{ C} \]

\[ \text{some} ::= \text{N D} - q2 \quad \text{person} ::= \text{N} \]

\[ \text{saw} ::= \text{D} = \text{D V} \]

\[ \text{every} ::= \text{N D} - q1 \quad \text{thing} ::= \text{N} \]

\[ (\text{saw}^\text{eet}_{x_1}, \text{every thing}, \epsilon) \]

To make the VP easier to point to, I put subject first, but as usual the selected subj is the 2nd arg of \textit{em}.\]
Example 1a:

\[
\epsilon ::= V +q1 +q2 \; C \quad \leftarrow (\text{saw}x_1 x_2, \text{every thing}, \text{some person}) \\
\text{some} ::= \text{N} \; \text{D} -q2 \quad \text{person} ::= \text{N} \\
\text{saw} ::= \text{D} = \text{D} \; V \\
\text{every} ::= \text{N} \; \text{D} -q1 \quad \text{thing} ::= \text{N}
\]

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:

\[ \epsilon ::= V +q1 +q2 C \quad \leftarrow (\text{saw}x_1 x_2, \text{every thing}, \text{some person}) \]

\[ \text{some} ::= N D -q2 \quad \text{person} ::= N \]

\[ \text{saw} ::= D =D V \]

\[ \text{every} ::= N D -q1 \quad \text{thing} ::= N \]

To make the VP easier to point to, I put subject first, but as usual the selected subj is the 2nd arg of \emph{em}.)
Example 1a:

\[ \iff \big( \text{some person} (\lambda x_2. \text{every thing} (\lambda x_1. \text{saw} x_1 x_2)), \epsilon, \epsilon \big) \]
\[ \iff \big( \text{every thing} (\lambda x_1. \text{saw} x_1 x_2), \epsilon, \text{some person} \big) \]

\[ \epsilon ::= V +q1 +q2 C \iff \big( \text{saw} x_1 x_2, \text{every thing}, \text{some person} \big) \]

\[ \text{some} ::= N D -q2 \quad \text{person} ::= N \]
\[ \text{saw} ::= D =D V \]
\[ \text{every} ::= N D -q1 \quad \text{thing} ::= N \]

To make the VP easier to point to, I put subject first, but as usual the selected subj is the 2nd arg of \textit{em}.
Example 1b:

\[
\epsilon ::= V +q2 +q1 C
\]

\[
\text{some} ::= N D -q2 \quad \text{person} ::= N
\]

\[
\text{saw} ::= D =D V
\]

\[
\text{every} ::= N D -q1 \quad \text{thing} ::= N
\]
Example 1b:

\[
\epsilon ::= V + q_2 + q_1 C \quad \leftarrow \quad (\text{saw}x_1, \text{every \ thing}, \epsilon)
\]

\[
\text{some} ::= \text{N D -}q_2 \quad \text{person} ::= \text{N} \\
\text{saw} ::= \text{D =D V} \\
\text{every} ::= \text{N D -}q_1 \quad \text{thing} ::= \text{N}
\]
Example 1b:

\[
\begin{align*}
\epsilon &::= V + q_2 + q_1 C \\
\text{some} &::= \mathbb{N} \ D - q_2 \ \text{person}::\mathbb{N} \\
\text{saw} &::= \mathbb{D} = D \ V \\
\text{every} &::= \mathbb{N} \ D - q_1 \ \text{thing}::\mathbb{N}
\end{align*}
\]

\[\leftarrow (\text{saw}_1 x_2, \text{every thing}, \text{some person})\]

\[\leftarrow (\text{saw}_1, \text{every thing}, \epsilon)\]
Example 1b:

\[ \epsilon ::= V +q2 +q1 C \quad \leftarrow (\text{some person}(\lambda x_2.\text{saw} x_1 x_2), \text{every thing}, \epsilon) \]

\[ \epsilon ::= V +q2 +q1 C \quad \leftarrow (\text{saw} x_1 x_2, \text{every thing}, \text{some person}) \]

\[ \epsilon ::= V +q2 +q1 C \quad \leftarrow (\text{saw} x_1, \text{every thing}, \epsilon) \]

\[ \text{some} ::= \text{D} -q2 \text{ person} ::= \text{N} \]

\[ \text{saw} ::= \text{D} =\text{D} V \]

\[ \text{every} ::= \text{D} -q1 \text{ thing} ::= \text{N} \]
Example 1b:

\[ \leftarrow \text{every thing}(\lambda x_1.\text{some person}(\lambda x_2.\text{saw} x_1 x_2)), \epsilon, \epsilon \] 

\[ \leftarrow \text{some person}(\lambda x_2.\text{saw} x_1 x_2), \text{every thing}, \epsilon \] 

\[ \epsilon ::= V +q_2 +q_1 C \] 

\[ \text{some} ::= N D -q_2 \text{ person} ::= N \] 

\[ \text{saw} ::= D =D V \] 

\[ \text{every} ::= N D -q_1 \text{ thing} ::= N \]
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

\( \epsilon ::= T +q2 C \)
\( \epsilon ::= V +k +\text{top } T \)

\( \text{some ::= } N \ D -k -q2 \)
\( \text{person ::= } N \)
\( \text{saw ::= } D =D V -\text{top} \)
\( \text{every ::= } N \ D \)
\( \text{thing ::= } N \)

\( \Leftarrow \) what could this be?
Hiraiwa’02: Someone saw everyone \((\exists > \forall, \forall > \exists)\)
Saw everyone, someone did \((\exists > \forall, \ast \forall > \exists)\)

\(\ast \forall > \exists\) if SpIC or other constraint blocks q-movement from spec, TP

Diagram:

\[ \epsilon ::= T +q2 C \]
\[ \epsilon ::= V +k +\text{top } T \]
\[ \text{some} ::=N D -k -q2 \quad \text{person} ::=N \quad \text{saw} ::=D =D \quad \text{top} \]
\[ \text{every} ::=N D \quad \text{thing} ::=N \]

\(\leftarrow (every \text{ thing } saw), \epsilon \)
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

\[
\begin{align*}
\epsilon &::= T + q2 C \\
\epsilon &::= V + k + \text{top } T \\
\text{some} &::= N \text{ D } - k - q2 \\
\text{person} &::= N \\
\text{see} &::= D \text{ =D V -top} \\
\text{every} &::= N \text{ D} \\
\text{thing} &::= N
\end{align*}
\]
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

\[
\leftarrow (\text{some person}(\lambda x_1. (\text{every thing saw}) x_1), \epsilon)
\]

\[
\epsilon::=T +q2\ C
\]

\[
\epsilon::=V +k +\text{top}\ T
\]

\[
\text{some}::=N\ D -k -q2\quad \text{person}::=N\quad \text{saw}::=D =D\ V -\text{top}
\]

\[
\text{every}::=N\ D\quad \text{thing}::=N
\]
Consider MG with some subset of features \( L = \{-f_1, \ldots, -f_k\} \) (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 \( \epsilon \)

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 \( i_i \) 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 \( f \)
Appendix: semantics for MGs

Example: checking

Improvements

For $t_1 [c] = a$ with $[a] = (s_0, \ldots, s_k)$, and $t_2 [c] = b$ with $[b] = (r_0, \ldots, r_k)$,

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

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

$$\begin{align*}
im(a) &= \begin{cases} 
[a] & \text{if } FF(t_2) = -f_i, i = j \quad \text{(ck)} \\
[a]_{0 : \text{some}(\lambda x_j. x_i = x_j \land s_0), i := s_j, j := \epsilon} & \text{if } FF(t_2) = -f_i \in L \quad \text{(ck)} \\
[a] & \text{if } FF(t_2) \not\in L \quad \text{(0)} \\
[a]_{j := \epsilon, 0 : s_j (\lambda x_j. s_0)} & \text{if } t_2 \text{ has no features.} \quad \text{(bnd)}
\end{cases}
\end{align*}$$

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

$\epsilon ::= V + q1 + q2 \ C$

Mary :: D $\iff$ what could this be?

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:

\[ \epsilon ::= V +q1 +q2 C \]

\[ \text{Mary} ::= D \quad \leftarrow (\text{saw}_{x_1}, \text{every \ thing}, \epsilon) \]

\[ \text{saw} ::= D =D V \]

\[ \text{every} ::= N \quad D -q1 -q2 \quad \text{thing} ::= N \]

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

\[ \epsilon ::= \text{V} +q1 +q2 \text{ C} \quad \leftarrow (\text{saw}_1 \text{mary, every thing}, \epsilon) \]

\[ \text{Mary} ::= \text{D} \quad \leftarrow (\text{saw}_1, \text{every thing}, \epsilon) \]

\[ \text{saw} ::= \text{D} =\text{D} \text{ V} \quad \leftarrow (\text{em}) \]

\[ \text{every} ::= \text{N} \text{ D} -q1 -q2 \quad \text{thing} ::= \text{N} \]

Again, to make the VP easier to point to, I put subject first, but selected subj is the 2nd arg of \text{em}.}
Appendix: semantics for MGs

Example: checking

Improvements

checking example:

\[
\begin{align*}
\epsilon ::= & V +q1 +q2 C \quad \leftarrow \, (\text{some}(\lambda x_1.x_1 = x_2 \land \text{saw}x_1\text{mary}), \epsilon, \text{every thing}) \\
\text{Mary} ::= & D \quad \leftarrow \, (\text{saw}x_1\text{mary}, \text{every thing}, \epsilon) \\
\text{saw} ::= & D =D V \quad \leftarrow \, (\text{saw}x_1, \text{every thing}, \epsilon) \\
\text{every} ::= & N D -q1 -q2 \quad \text{thing} ::= N
\end{align*}
\]

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

\[ \text{ every thing}(\lambda x_2.\text{some}(\lambda x_1.x_1 = x_2 \land \text{saw}x_1\text{mary})), \epsilon, \epsilon) \]

\[ \text{ some}(\lambda x_1.x_1 = x_2 \land \text{saw}x_1\text{mary}), \epsilon, \text{ every thing} \]

\[ \epsilon::=V +q1 +q2 \text{ C} \]

\[ \text{ saw}::=D =D V \]

\[ \text{ every}::=N D -q1 -q2 \]

\[ \text{ thing}::=N \]

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 \([\text{move}]\) uniform?
  yes, and this gets us closer to 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
"spell out"
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  \rightarrow  the king eats the pie
the king have -s eat -en the pie  \rightarrow  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  $\mapsto$  the king eats the pie
the king have -s eat -en the pie  $\mapsto$  the king has eaten the pie
the king have -s laugh -en  $\mapsto$  the king has laughed

(CL: Roark&Sproat’07,Huet’03)
(Ph: Riggle’04,Eisner’97)
Morphophonology as transduction

the king eat -s the pie  \rightarrow \text{the king eats the pie}
the king have -s eat -en the pie \rightarrow \text{the king has eaten the pie}
the king have -s laugh -en \rightarrow \text{the king has laughed}
the king be -s laugh -ing \rightarrow \text{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 have -s eat -en the pie
the king have -s laugh -en
the king be -s laugh -ing
the king will -s laugh

:Boolean: Roark&Sproat’07; Huet’03
(Ph: Riggle’04; Eisner’97)
Morphophonology as transduction

the king eat -s the pie  \rightarrow  the king eats the pie
the king have -s eat -en the pie  \rightarrow  the king has eaten the pie
the king have -s laugh -en  \rightarrow  the king has laughed
the king be -s laugh -ing  \rightarrow  the king’s laughing
the king will -s laugh  \rightarrow  the king will laugh
-s the king laugh  \rightarrow  does the king laugh

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

the king eat -s the pie $\mapsto$ the king eats the pie
forward:
the king eat -s the pie $\leftrightarrow$ the king eats the pie

backward:
the king eats the pie $\leftrightarrow$ 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 \( \approx 7 \) 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

For $t_1[c] = a$ with $[a] = (s_0, \ldots, s_k)$, and $t_2[c] = b$ with $[b] = (r_0, \ldots, r_k)$,

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

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

$$\boxed{\text{im}(a) = \begin{cases} [a] & \text{if } FF(t_2) = -f_i, i = j \quad (\text{ck}) \\ [a][0:=\text{some}(\lambda x_j.x_i=x_j \land s_0), i:=s_j, j:=\epsilon] & \text{if } FF(t_2) = -f_i \in L \quad (\text{ck}) \\ [a] & \text{if } FF(t_2) \not\in L \quad (0) \\ [a][j:=\epsilon, 0:=s_j(\lambda x_j.s_0)] & \text{if } t_2 \text{ has no features.} \quad (\text{bnd}) \end{cases}}$$
References


---

E Stabler, UCLA

Grammar in Performance and Acquisition:interfaces