LF-Interpretation, Compositionally

Greg Kobele

University of Chicago

Dec 02

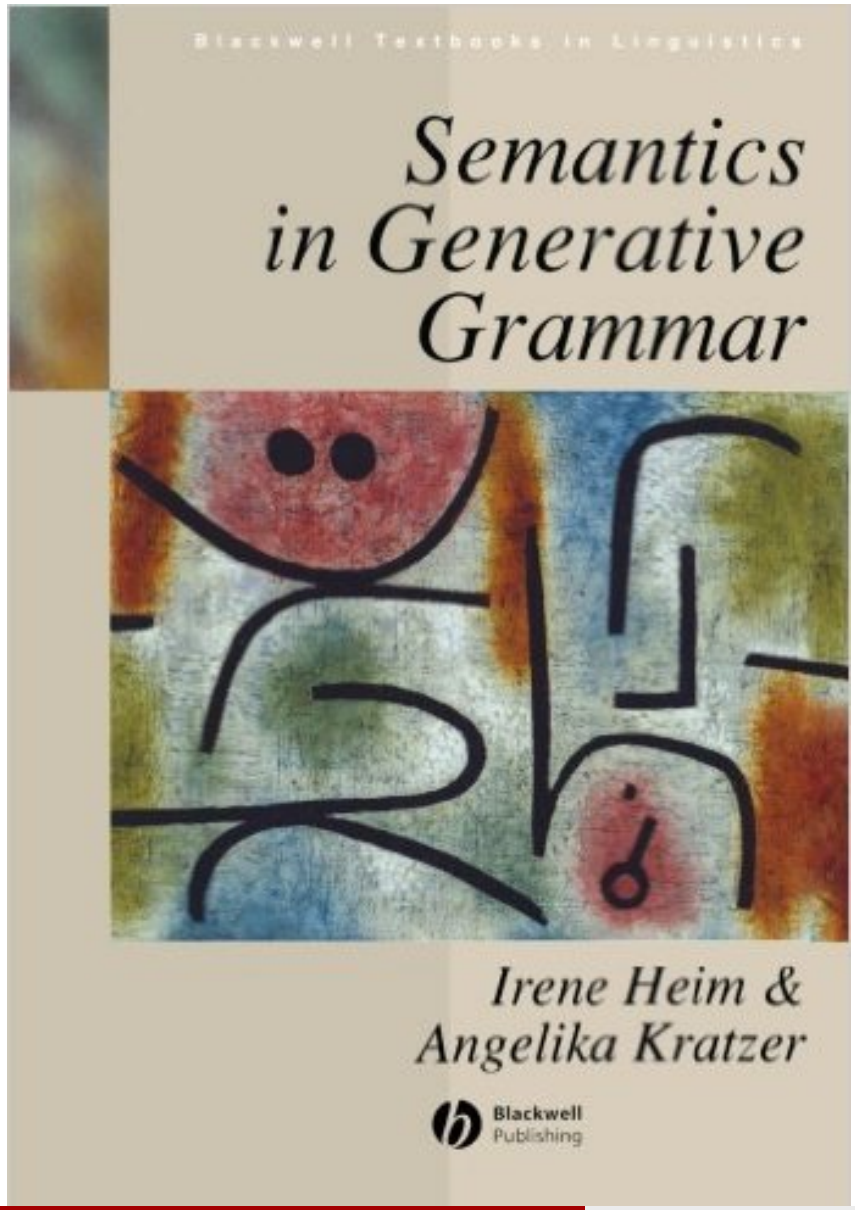
Plan

- Give a compositional semantics for minimalism

Main claim

LF-interpretation can be viewed directly compositionally

Semantics in Generative Grammar



- Binary branching nodes

$$\left[\!\!\left[\begin{array}{ccc} & \bullet & \\ / & & \backslash \\ \alpha & & \beta \end{array} \right]\!\!\right]^g = [\![\alpha]\!]^g \oplus [\![\beta]\!]^g$$

- Unary branching nodes

$$\left[\!\!\left[\begin{array}{c} \bullet \\ | \\ \alpha \end{array} \right]\!\!\right]^g = [\![\alpha]\!]^g$$

- Binding

$$\left[\!\!\left[\begin{array}{ccc} & \bullet & \\ / & & \backslash \\ i & & \alpha \end{array} \right]\!\!\right]^g = \lambda x.[\![\alpha]\!]^{g[i:=x]}$$

- Traces

$$[\![t_i]\!]^g = g(i)$$

Interpreting λ terms in type structures

- Application
  $[\![(M\ N)]\!]^g = [\![M]\!]^g\ ([\![N]\!]^g)$

- Abstraction
  $[\![\lambda i.M]\!]^g = \lambda\!\!\lambda x.[\![M]\!]^{g[i:=x]}$

- Variables
  $[\![i]\!]^g = g(i)$

- Binary branching nodes
  $$\left[\!\!\left[\begin{array}{ccc} & \bullet & \\ / & & \backslash \\ \alpha & & \beta \end{array}\right]\!\!\right]^g = [\![\alpha]\!]^g \oplus [\![\beta]\!]^g$$

- Unary branching nodes
  $$\left[\!\!\left[\begin{array}{c} \bullet \\ | \\ \alpha \end{array}\right]\!\!\right]^g = [\![\alpha]\!]^g$$

- Binding
  $$\left[\!\!\left[\begin{array}{ccc} & \bullet & \\ / & & \backslash \\ i & & \alpha \end{array}\right]\!\!\right]^g = \lambda\!\!\lambda x.[\![\alpha]\!]^{g[i:=x]}$$

- Traces
  $[\![t_i]\!]^g = g(i)$

Parts and their meanings

Most expressions don't have any meaning

$$\left[\!\left[\begin{array}{c} \text{V} \\ \text{V} \quad \text{D} \\ \text{praise} \quad \text{D} \quad \text{N} \\ \text{every} \quad \text{boy} \end{array} \right]\!\right]^g = [\![\mathit{praise}]\!]^g \oplus \left[\!\left[\begin{array}{c} \text{D} \\ \text{D} \quad \text{N} \\ \text{every} \quad \text{boy} \end{array} \right]\!\right]^g$$

$$= [\![\mathit{praise}]\!]^g \oplus ([\![\mathit{every}]\!]^g \oplus [\![\mathit{boy}]\!]^g)$$

- $[\![\mathit{every}]\!]^g \oplus [\![\mathit{boy}]\!]^g : (et)t$
- $[\![\mathit{praise}]\!]^g : eet$

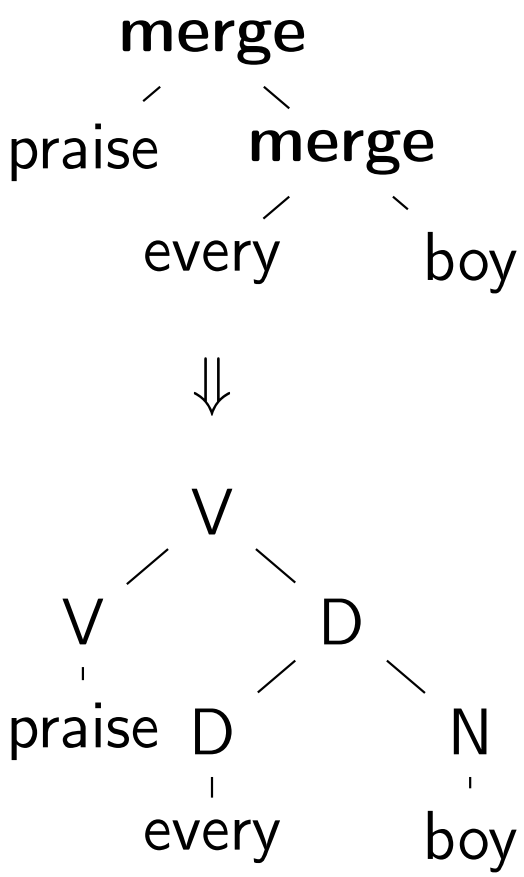
these cannot be combined!

FA $\alpha\beta \to \alpha \to \beta$

BA $\alpha \to \alpha\beta \to \beta$

PM $\alpha t \to \alpha t \to \alpha t$

Revisiting meaningless parts



What is the contribution of *praise every boy* to expressions it is part of?

a quantifier part **every**(**boy**)($\lambda x. \ldots$

and a property part **praise**(x)

Let's write instead:

$$[\textbf{every}(\textbf{boy})]_x \vdash \textbf{praise}(x)$$

Notation and Operations

$$[\textbf{every}(\textbf{boy})]_x \vdash \textbf{praise}(x)$$

The general case, with multiple stored quantifiers:

$$[Q_1]_{x_1}, \ldots, [Q_i]_{x_i} \vdash M$$

The entire point is to ignore what is stored

$$\frac{M}{\vdash M} \uparrow \qquad \frac{\Gamma \vdash M \qquad \Delta \vdash N}{\Gamma, \Delta \vdash M\ N} <*>$$

Working with Storage

$$\dfrac{\dfrac{\textbf{seem}}{\vdash \textbf{seem}}\uparrow \qquad \dfrac{\dfrac{\textbf{Pass}}{\vdash \textbf{Pass}}\uparrow \qquad \begin{array}{c}\vdots\\ [\textbf{every}(\textbf{boy})]_x \vdash \textbf{praise}(x)\end{array}}{[\textbf{every}(\textbf{boy})]_x \vdash \textbf{Pass}(\textbf{praise}(x))}\,\text{<*>}}{[\textbf{every}(\textbf{boy})]_x \vdash \textbf{seem}(\textbf{Pass}(\textbf{praise}(x)))}\,\text{<*>}$$

Building *praise every boy*

$$\dfrac{\dfrac{\textbf{praise}}{\vdash \textbf{praise}}\uparrow \qquad \dfrac{\dfrac{\textbf{every}}{\vdash \textbf{every}}\uparrow \qquad \dfrac{\textbf{boy}}{\vdash \textbf{boy}}\uparrow}{\vdash \textbf{every boy}}\,\text{<*>}}{\text{type mismatch!}}\,\text{<*>}$$

Building *praise every boy*

$$\dfrac{\textbf{praise}}{\vdash \textbf{praise}}\uparrow \qquad \dfrac{\dfrac{\textbf{every}}{\vdash \textbf{every}}\uparrow \quad \dfrac{\textbf{boy}}{\vdash \textbf{boy}}\uparrow}{\vdash \textbf{every boy}}\,\text{<*>}$$

We want to 'insert a trace'

$$\dfrac{\vdash M}{[M]_x \vdash x}\,\Box$$

Building *praise every boy*

$$\dfrac{\textbf{praise}}{\vdash \textbf{praise}}\uparrow \qquad \dfrac{\dfrac{\dfrac{\textbf{every}}{\vdash \textbf{every}}\uparrow \quad \dfrac{\textbf{boy}}{\vdash \textbf{boy}}\uparrow}{\vdash \textbf{every boy}}\text{<*>}}{[\textbf{every boy}]_x \vdash x}\,\Box$$

We want to 'insert a trace'

$$\dfrac{\vdash M}{[M]_x \vdash x}\,\Box$$

Building *praise every boy*

$$\dfrac{\dfrac{\textbf{praise}}{\vdash \textbf{praise}}\uparrow \qquad \dfrac{\dfrac{\dfrac{\textbf{every}}{\vdash \textbf{every}}\uparrow \qquad \dfrac{\textbf{boy}}{\vdash \textbf{boy}}\uparrow}{\vdash \textbf{every boy}}\,{<*>}}{[\textbf{every boy}]_x \vdash x}\,\Box}{[\textbf{every boy}]_x \vdash \textbf{praise}\ x}\,{<*>}$$

We want to 'insert a trace'

$$\dfrac{\vdash M}{[M]_x \vdash x}\,\Box$$

Taking things out of storage

$$\dfrac{\dfrac{\mathbf{seem}}{\vdash \mathbf{seem}}\uparrow \qquad \dfrac{\dfrac{\mathbf{Pass}}{\vdash \mathbf{Pass}}\uparrow \qquad \begin{array}{c}\vdots\\ [\mathbf{every}(\mathbf{boy})]_x \vdash \mathbf{praise}(x)\end{array}}{[\mathbf{every}(\mathbf{boy})]_x \vdash \mathbf{Pass}(\mathbf{praise}(x))}\,{<*>}}{[\mathbf{every}(\mathbf{boy})]_x \vdash \mathbf{seem}(\mathbf{Pass}(\mathbf{praise}(x)))}\,{<*>}$$

Taking things out of storage

$$\dfrac{\dfrac{\mathbf{seem}}{\vdash \mathbf{seem}}\uparrow \qquad \dfrac{\dfrac{\mathbf{Pass}}{\vdash \mathbf{Pass}}\uparrow \qquad \begin{array}{c}\vdots\\ [\mathbf{every}(\mathbf{boy})]_x \vdash \mathbf{praise}(x)\end{array}}{[\mathbf{every}(\mathbf{boy})]_x \vdash \mathbf{Pass}(\mathbf{praise}(x))}\,<*>}{[\mathbf{every}(\mathbf{boy})]_x \vdash \mathbf{seem}(\mathbf{Pass}(\mathbf{praise}(x)))}\,<*>$$

retrieval

$$\dfrac{\Gamma, [M_i]_{x_i}, \Delta \vdash N}{\Gamma, \Delta \vdash M_i \oplus (\lambda x_i.N)}\,\langle\cdot\rangle^i_{\oplus}$$

Taking things out of storage

$$\dfrac{\dfrac{\dfrac{\textbf{seem}}{\vdash \textbf{seem}}\uparrow \qquad \dfrac{\dfrac{\textbf{Pass}}{\vdash \textbf{Pass}}\uparrow \qquad \begin{array}{c}\vdots\\ [\textbf{every}(\textbf{boy})]_x \vdash \textbf{praise}(x)\end{array}}{[\textbf{every}(\textbf{boy})]_x \vdash \textbf{Pass}(\textbf{praise}(x))}\,\text{<*>}}{[\textbf{every}(\textbf{boy})]_x \vdash \textbf{seem}(\textbf{Pass}(\textbf{praise}(x)))}\,\text{<*>}}{\vdash \textbf{every}(\textbf{boy})(\lambda x.\textbf{seem}(\textbf{Pass}(\textbf{praise}(x))))}\,\langle\cdot\rangle^{1}_{\mathtt{FA}}$$

retrieval

$$\dfrac{\Gamma, [M_i]_{x_i}, \Delta \vdash N}{\Gamma, \Delta \vdash M_i \oplus (\lambda x_i.N)}\,\langle\cdot\rangle^{i}_{\oplus}$$

Manipulating Stores

pure

$$\frac{M}{\vdash M} \uparrow$$

apply

$$\frac{\Gamma \vdash M \qquad \Delta \vdash N}{\Gamma, \Delta \vdash M\ N} <*>$$

retrieve

$$\frac{\Gamma, [M_i]_{x_i}, \Delta \vdash N}{\Gamma, \Delta \vdash M_i \oplus (\lambda x_i.N)} \langle\cdot\rangle^i_\oplus$$

store

$$\frac{\vdash M}{[M]_x \vdash x} \Box$$

Understanding stores

$$[M_1]_{x_1}, \ldots, [M_i]_{x_i} \vdash N$$
$$\Rightarrow \lambda k.k\ M_1\ \ldots\ M_i\ (\lambda x_1, \ldots, x_i.N)$$

Example

$$[\textbf{every boy}]_x \vdash \textbf{praise}\ x$$
$$\Rightarrow \lambda k.k\ (\textbf{every boy})\ (\lambda x.\textbf{praise}\ x)$$

Some examples

pure

$$\frac{M}{\vdash M}\uparrow$$

$$\Downarrow$$

$$\frac{M}{\lambda k.k\ M}\uparrow$$

storage

$$\frac{\vdash M}{[M]_x \vdash x}\Box$$

$$\Downarrow$$

$$\frac{\lambda k.k\ M}{\lambda k.k\ M\ (\lambda x.x)}\Box$$

$$M^{\uparrow} \equiv \lambda k.k\ M$$

$$\Box m \equiv \lambda k.m\ (\lambda M.k\ M\ (\lambda x.x))$$

More notation

idiom brackets

write $(\!| f\ a_1\ \ldots\ a_i |\!)$

for $f^{\uparrow}$ <*> a_1 <*> ... <*> a_i

application

Forward $f \rhd a := f\ a$

Backward $a \lhd f := f\ a$

Minimalist semantics

$$[\![\textbf{merge}]\!] \mapsto \lambda m, n.(\!| m \oplus n |\!)$$
$$[\![\textbf{merge}]\!] \mapsto \lambda m, n.(\!| m \oplus \Box n |\!)$$

$$[\![\textbf{move}]\!] \mapsto \lambda m.m$$
$$[\![\textbf{move}]\!] \mapsto \lambda m.\langle m \rangle^{k}_{\oplus}$$

$$[\![\ell]\!] = \mathcal{I}(\ell)^{\uparrow}$$

for $\oplus \in \{\triangleright, \triangleleft\}$

Unpacking the notation

Recall that

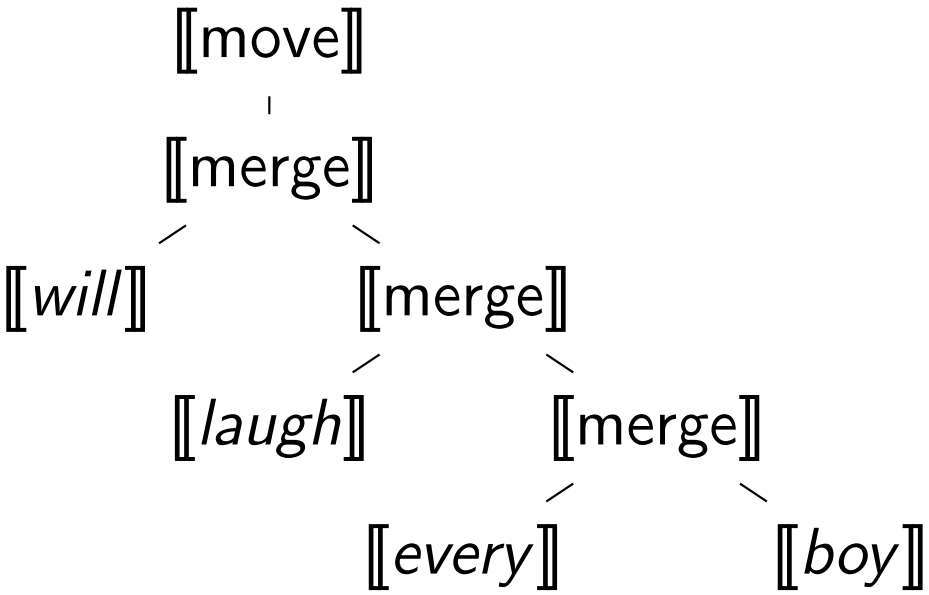
$$\lambda m, n.(\!| m \rhd n |\!)$$

means

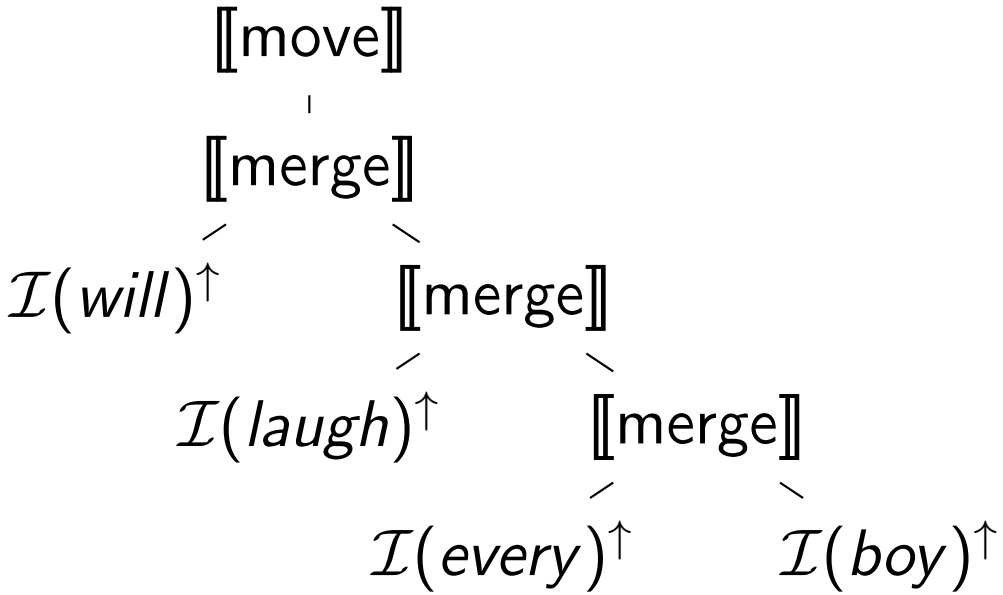
$$\lambda m, n.(\rhd)^{\uparrow} <*> m <*> n$$

$$\dfrac{\dfrac{\dfrac{\rhd}{\vdash \rhd}\uparrow \qquad \overset{(m)}{\Gamma \vdash M}}{\Gamma \vdash M \rhd}<*> \qquad \overset{(n)}{\Delta \vdash N}}{\Gamma, \Delta \vdash M \rhd N}<*>$$

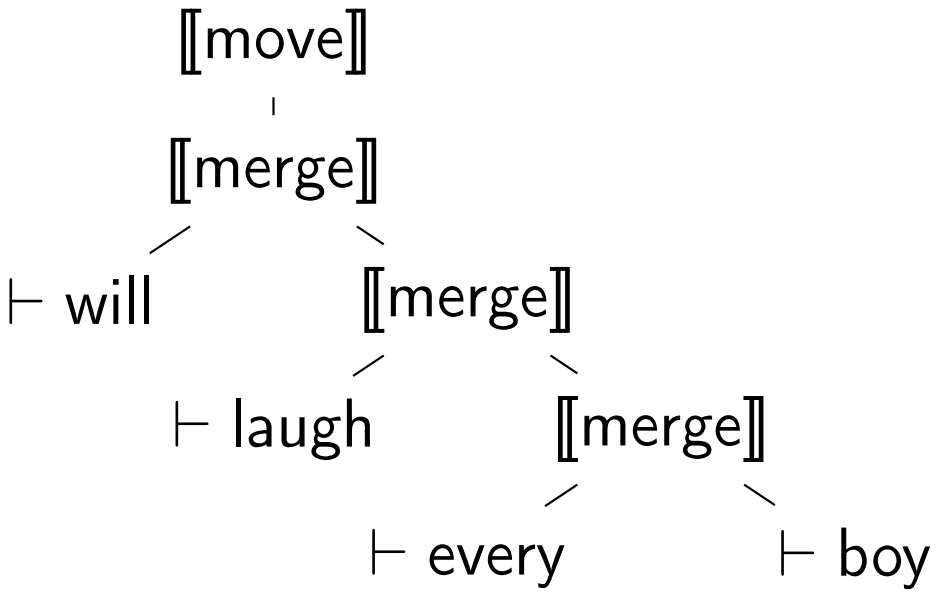
Every boy laughs



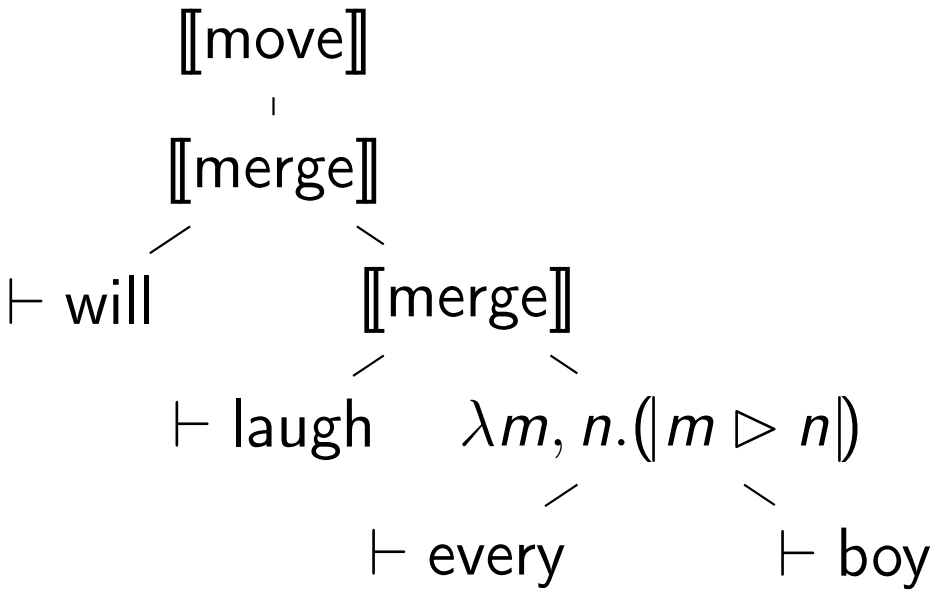
Every boy laughs



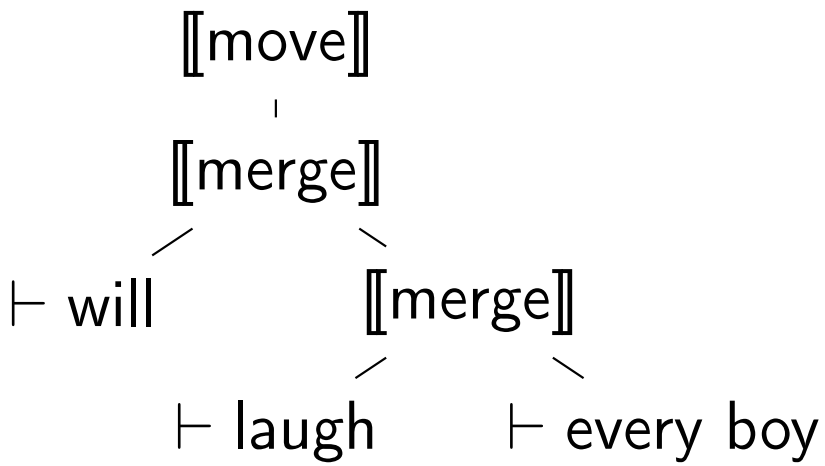
Every boy laughs



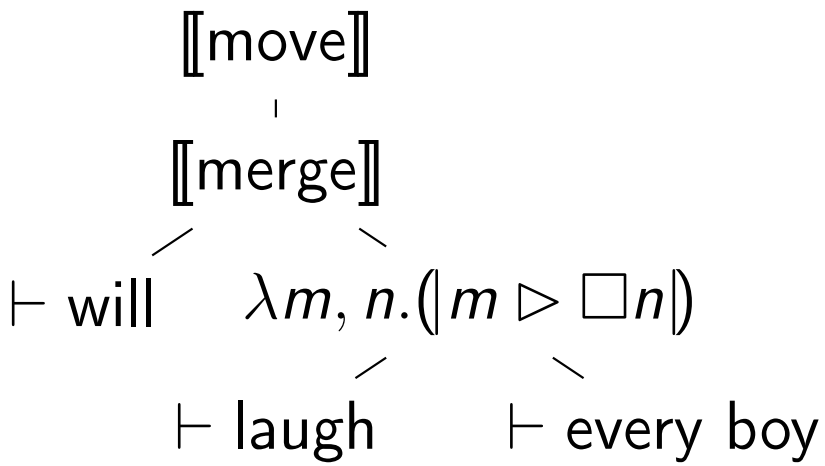
Every boy laughs



Every boy laughs



Every boy laughs



Every boy laughs

$$
\begin{tikzcd}
 & [\![\text{move}]\!] \arrow[d, dash] & \\
 & [\![\text{merge}]\!] \arrow[dl, dash] \arrow[dr, dash] & \\
\vdash \text{will} & & [\text{every boy}]_x \vdash \text{laugh } x
\end{tikzcd}
$$

Every boy laughs

$$\llbracket \text{move} \rrbracket$$

$$\lambda m, n.(\!| m \rhd n |\!)$$

$$\vdash \text{will} \qquad [\text{every boy}]_x \vdash \text{laugh } x$$

Every boy laughs

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

$$[\text{every boy}]_x \vdash \text{will (laugh } x)$$

Every boy laughs

$$\lambda m.\langle m\rangle^{1}_{\triangleright}$$

$$[\text{every boy}]_x \vdash \text{will (laugh } x)$$

Every boy laughs

$$\vdash \text{every boy } (\lambda x.\text{will } (\text{laugh } x))$$

Plan

- Present algebraic laws of cooper storage
- Introduce delimited continuations

Main claim

LF-interpretation à la H&K is based on delimited continuations

Applicative Functor Laws

identity

$$\mathtt{id}^{\uparrow} \mathbin{<*>} u = u$$

composition

$$((\circ^{\uparrow} \mathbin{<*>} u) \mathbin{<*>} v) \mathbin{<*>} w = u \mathbin{<*>} (v \mathbin{<*>} w)$$

homomorphism

$$f^{\uparrow} \mathbin{<*>} x^{\uparrow} = (f\ x)^{\uparrow}$$

interchange

$$u \mathbin{<*>} x^{\uparrow} = (\lambda P.Px)^{\uparrow} \mathbin{<*>} u$$

Deriving *homomorphism*

homomorphism

$$f^{\uparrow} \mathbin{<*>} x^{\uparrow} = (f\ x)^{\uparrow}$$

$$\dfrac{\dfrac{f}{\vdash f}\uparrow \qquad \dfrac{x}{\vdash x}\uparrow}{\vdash f\ x}\,{<*>} \qquad = \qquad \dfrac{f\ x}{\vdash f\ x}\uparrow$$

Laws Particular to Cooper Storage

1. That which is stored, can be retrieved

$$\langle E[\Box_k M]\rangle_N^k = N^{\uparrow} <*> M <*> \langle E[\Box_k \mathtt{id}^{\uparrow}]\rangle_{\triangleright}^k$$

2. Storing and retrieving vacuity is vacuous

$$\langle u <*> \Box_k \mathtt{id}^{\uparrow}\rangle_{\triangleright}^k = u$$

Deriving Law 1

Retrieval from storage

$$\langle E[\Box_k M]\rangle^k_N = N^{\uparrow} \oplus M \oplus \langle E[\Box_k \mathtt{id}^{\uparrow}]\rangle^k_{\rhd}$$

$$\dfrac{\dfrac{\vdots}{\Gamma, [M]_{x_k}, \Delta \vdash O}}{\Gamma, \Delta \vdash N\ M\ (\lambda x_k.O)}\ \langle\cdot\rangle^k_N \quad = \quad \dfrac{\dfrac{\dfrac{N}{\vdash N}\uparrow \qquad \vdash M}{\vdash N\ M}\ \text{<*>} \qquad \dfrac{\dfrac{\dfrac{\mathtt{id}}{\vdash \mathtt{id}}\uparrow}{[\mathtt{id}]_{x_k} \vdash x_k}\ \Box_k \quad \vdots}{\Gamma, [\mathtt{id}]_{x_k}, \Delta \vdash O}}{\Gamma, \Delta \vdash \mathtt{id} \rhd (\lambda x_k.O)}\ \langle\cdot\rangle^k_{\rhd}}{\Gamma, \Delta \vdash \lambda x_k.O}\ \text{<*>}}{\Gamma, \Delta \vdash N\ M\ (\lambda x_k.O)}\ \text{<*>}$$

$$\dfrac{\vdash M}{[M]_{x_k} \vdash x_k}\ \Box_k$$

shift0 and reset

- x
- $(M\ N)$
- $\lambda x.M$
- $\Box_k M$
- $\langle M \rangle^k$

$$
\begin{array}{rl}
\beta & (\lambda x.M)\ N \rightsquigarrow M[x := N] \\
\eta & \lambda x.(M\ x) \rightsquigarrow M \\
\texttt{shift0} & \langle E[\Box_k M] \rangle^k \rightsquigarrow M\ (\lambda x.\langle E[x] \rangle^k) \\
\texttt{reset} & \langle V \rangle^k \rightsquigarrow V
\end{array}
$$

shift0 and reset

- x
- $(M\ N)$
- $\lambda x.M$
- $\Box_k M$
- $\langle M \rangle^k$

$$
\begin{array}{rl}
\beta & (\lambda x.M)\ N \rightsquigarrow M[x := N] \\
\eta & \lambda x.(M\ x) \rightsquigarrow M \\
\texttt{shift0} & \langle E[\Box_k M] \rangle^k \rightsquigarrow M\ (\lambda x.\langle E[x] \rangle^k) \\
\texttt{reset} & \langle V \rangle^k \rightsquigarrow V
\end{array}
$$

$$\langle E[\Box_k M] \rangle^k \rightsquigarrow M\ (\lambda x.\langle E[x] \rangle^k)$$

$$\langle E[\Box_k M] \rangle^k_{\triangleright} = M \texttt{<*>} \langle E[\Box_k \texttt{id}^{\uparrow}] \rangle^k_{\triangleright}$$

Minimalist semantics (via delimited continuations)

$$[\![\textbf{merge}]\!] \mapsto \lambda m, n.(\!| m \oplus n |\!)$$
$$[\![\textbf{merge}]\!] \mapsto \lambda m, n.(\!| m \oplus \Box n |\!)$$

$$[\![\textbf{move}]\!] \mapsto \lambda m.m$$
$$[\![\textbf{move}]\!] \mapsto \lambda m.\langle m \rangle^{k}_{\oplus}$$

$$[\![\ell]\!] = \mathcal{I}(\ell)^{\uparrow}$$

for $\oplus \in \{\triangleright, \triangleleft\}$

Minimalist semantics (via delimited continuations)

$$[\![\textbf{merge}]\!] \mapsto \lambda m, n.m \oplus n$$
$$[\![\textbf{merge}]\!] \mapsto \lambda m, n.(\!|m \oplus \Box n|\!)$$

$$[\![\textbf{move}]\!] \mapsto \lambda m.m$$
$$[\![\textbf{move}]\!] \mapsto \lambda m.\langle m \rangle^{k}_{\oplus}$$

$$[\![\ell]\!] = \mathcal{I}(\ell)$$

for $\oplus \in \{\triangleright, \triangleleft\}$

Minimalist semantics (via delimited continuations)

$$[\![\textbf{merge}]\!] \mapsto \lambda m, n.m \oplus n$$
$$[\![\textbf{merge}]\!] \mapsto \lambda m, n.(\!|m \oplus \Box n|\!)$$

$$[\![\textbf{move}]\!] \mapsto \lambda m.m$$
$$[\![\textbf{move}]\!] \mapsto \lambda m.\langle m \rangle^{k}_{\oplus}$$

$$[\![\ell]\!] = \mathcal{I}(\ell)$$

for $\oplus \in \{\triangleright, \triangleleft\}$

Minimalist semantics (via delimited continuations)

$$[\![\textbf{merge}]\!] \mapsto \lambda m, n.m \oplus n$$
$$[\![\textbf{merge}]\!] \mapsto \lambda m, n.m \oplus \Box_k(\lambda o.n \oplus o)$$

$$[\![\textbf{move}]\!] \mapsto \lambda m.m$$
$$[\![\textbf{move}]\!] \mapsto \lambda m.\langle m \rangle^k$$

$$[\![\ell]\!] = \mathcal{I}(\ell)$$

for $\oplus \in \{\triangleright, \triangleleft\}$

Plan

- give examples of the benefits of viewing LF-interpretation compositionally

Main claim

LF-interpretation not only *can* be viewed directly compositionally, it *should* be.

Generative Capacity

The set of meanings assigned to sentences of a grammar is

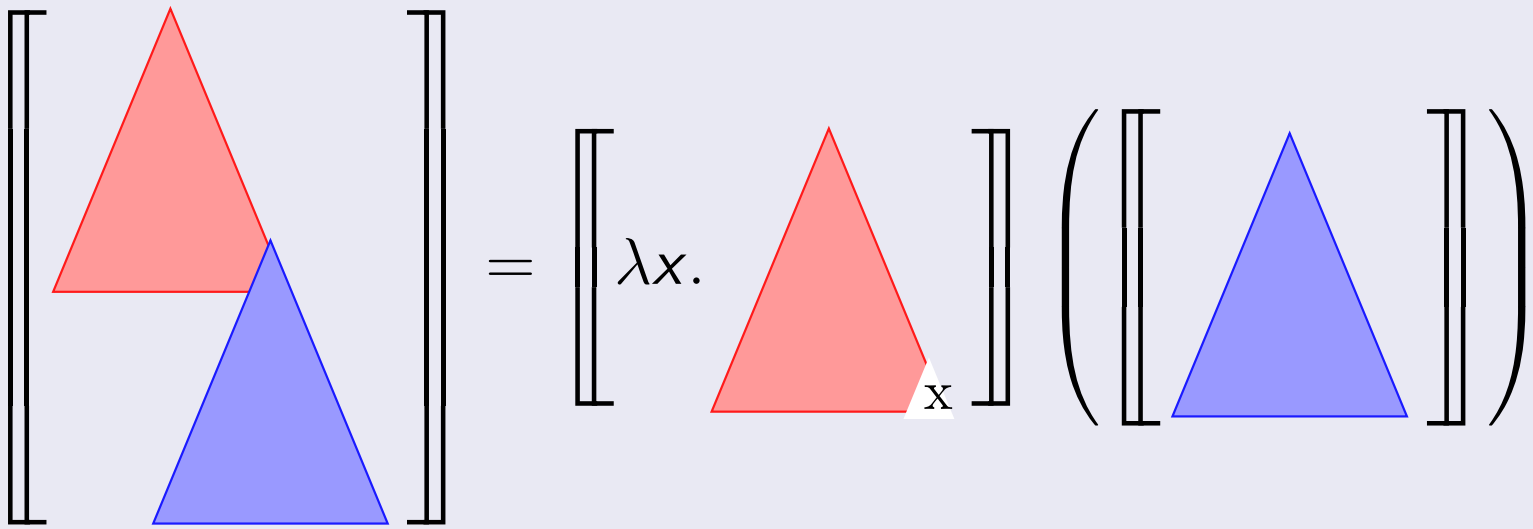
- a linear higher order IO language

and thus the set of structures which yield a given λ-term

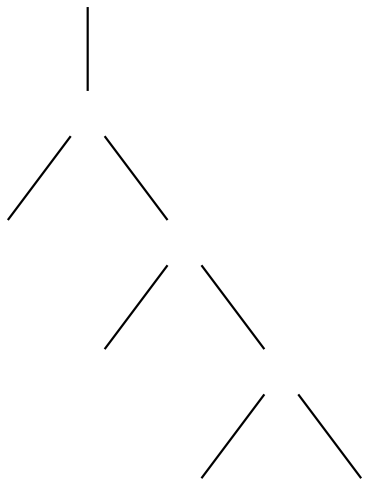
- can be found in polynomial time

Decompositionality

any part of a derivation has a normal semantic value



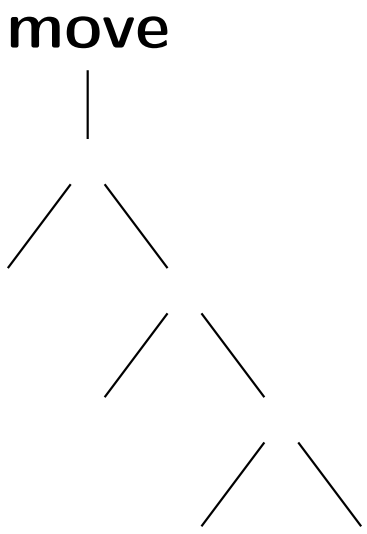
Top-down parsing



$\lambda m.$

m

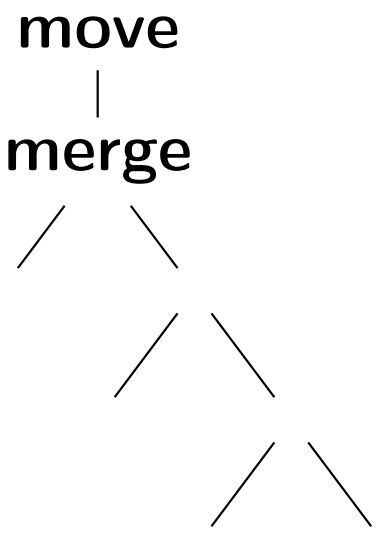
Top-down parsing



$\lambda m.$

$\langle m \rangle^{k}_{\triangleright}$

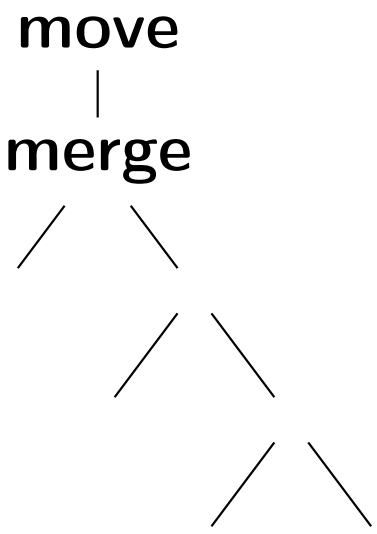
Top-down parsing



$$\lambda t, v.$$

$$\langle \triangleright^{\uparrow} <*> t <*> v \rangle^{k}_{\triangleright}$$

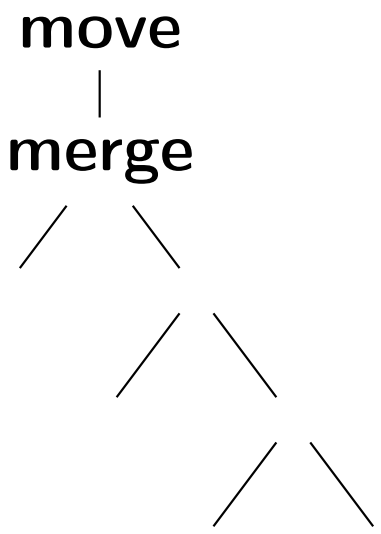
Top-down parsing



$$\lambda t, v.$$

$$\langle \triangleright^{\uparrow} \mathbin{<*>} t \mathbin{<*>} v \rangle^{k}_{\triangleright}$$

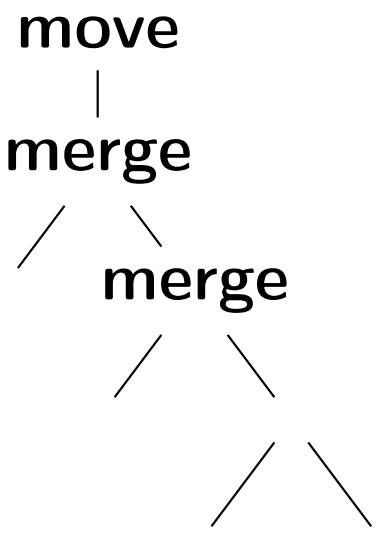
Top-down parsing



$$\lambda t, v.$$

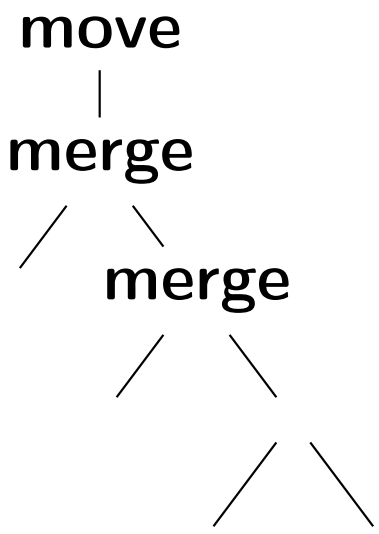
$$\langle t \text{ <*> } v \rangle^{k}_{\triangleright}$$

Top-down parsing



$$\lambda t, V, s.$$
$$\langle t <*> (\triangleright^{\uparrow} <*> V <*> \Box_k s)\rangle^k_{\triangleright}$$

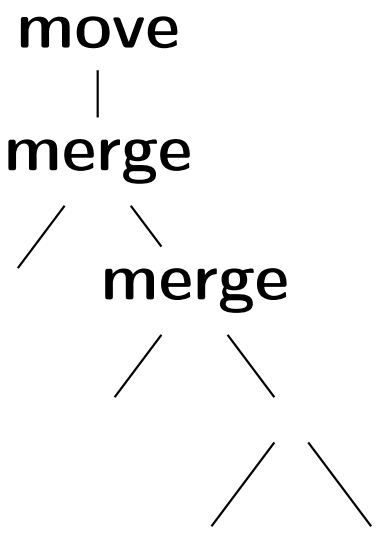
Top-down parsing



$$\lambda t, V, s.$$

$$\langle t <*> (\triangleright^{\uparrow} <*> V <*> \square_k s)\rangle^k_{\triangleright}$$

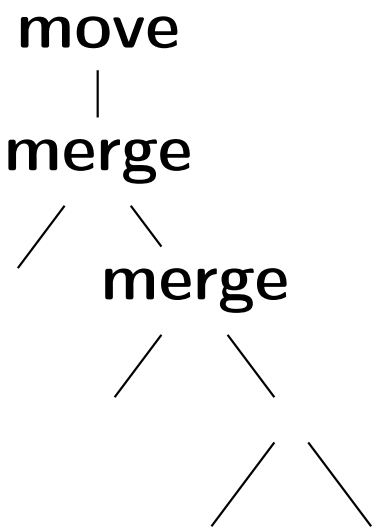
Top-down parsing



$$\lambda t, V, s.$$

$$\langle t <*> (V <*> \square_k s) \rangle^k_{\triangleright}$$

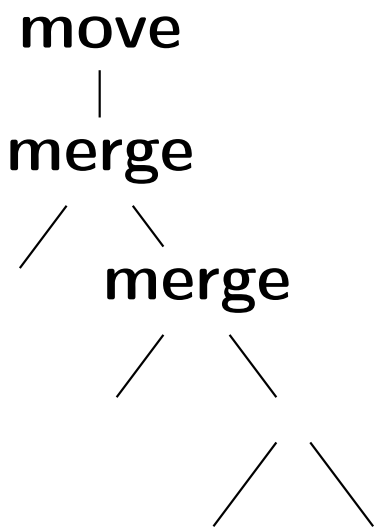
Top-down parsing



$$\lambda t, V, s.$$

$$\langle t \mathbin{<*>} (V \mathbin{<*>} \square_k s) \rangle^k_{\triangleright}$$

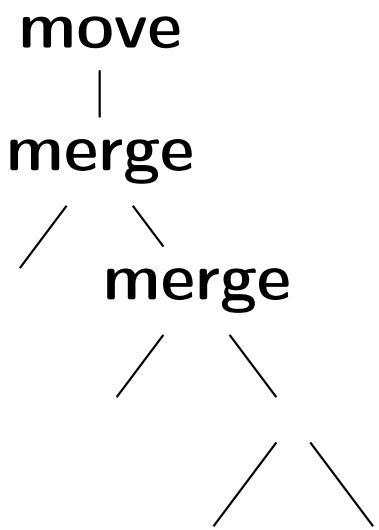
Top-down parsing



$$\lambda t, V, s.$$

$$s \mathbin{<*>} \langle t \mathbin{<*>} (V \mathbin{<*>} \Box_k \mathtt{id}^{\uparrow}) \rangle^k_{\triangleright}$$

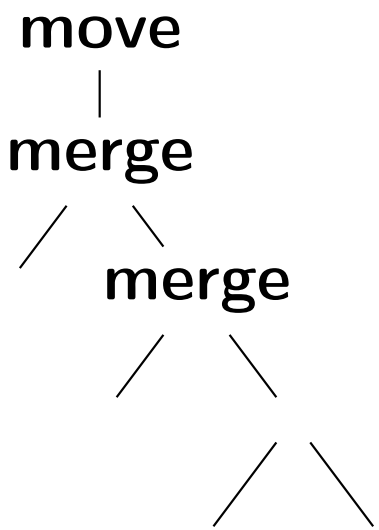
Top-down parsing



$$\lambda t, V, s.$$

$$s \mathbin{<*>} \langle t \mathbin{<*>} (V \mathbin{<*>} \Box_k \mathtt{id}^{\uparrow}) \rangle^k_{\triangleright}$$

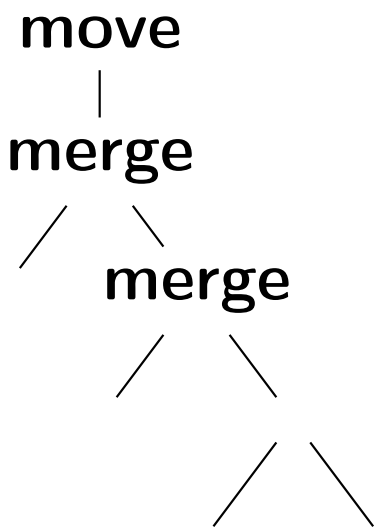
Top-down parsing



$$\lambda t, V, s.$$

$$s \mathbin{<*>} \langle (\circ^{\uparrow} \mathbin{<*>} t \mathbin{<*>} V) \mathbin{<*>} \Box_k \mathtt{id}^{\uparrow} \rangle^{k}_{\triangleright}$$

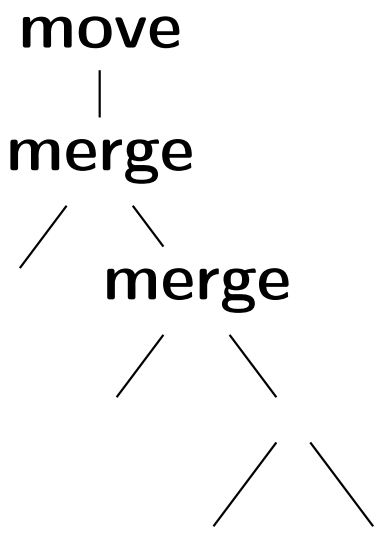
Top-down parsing



$$\lambda t, V, s.$$

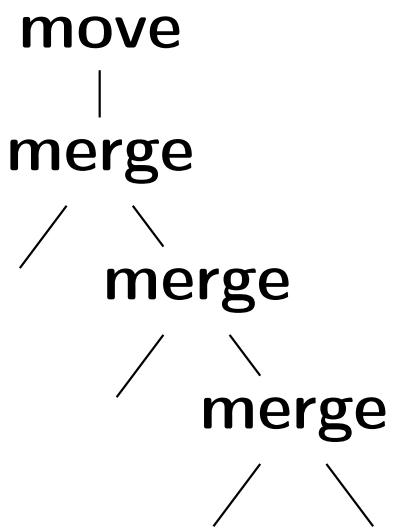
$$s \mathbin{<*>} \langle (\circ^{\uparrow} \mathbin{<*>} t \mathbin{<*>} V) \mathbin{<*>} \Box_k \mathtt{id}^{\uparrow} \rangle^{k}_{\triangleright}$$

Top-down parsing



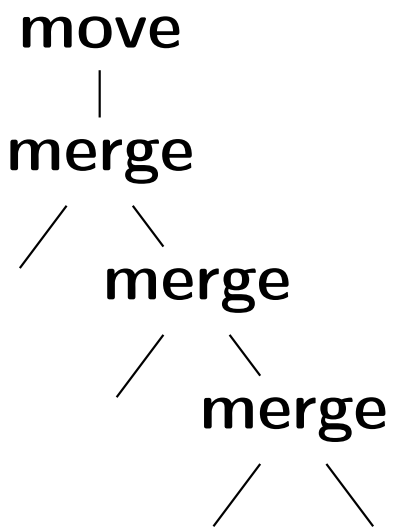
$$\lambda t, V, s.$$
$$s <*> (\circ^{\uparrow} <*> t <*> V)$$

Top-down parsing



$$\lambda t, V, d, n.$$
$$(\triangleright^{\uparrow} <*> d <*> n) <*> (\circ^{\uparrow} <*> t <*> V)$$

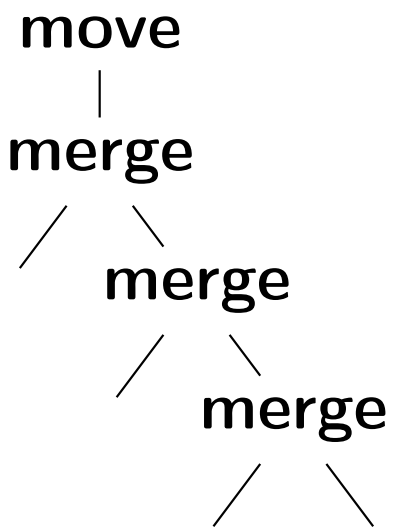
Top-down parsing



$$\lambda t, V, d, n.$$

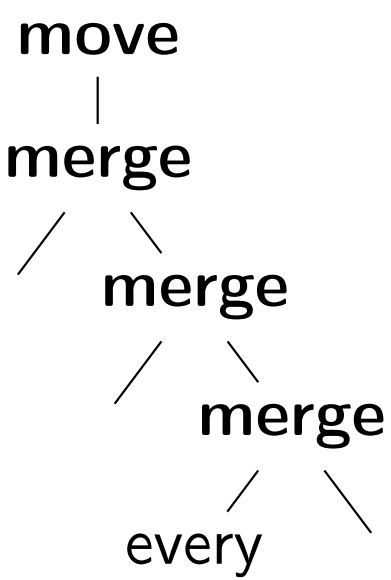
$$(\triangleright^{\uparrow} <*> d <*> n) <*> (\circ^{\uparrow} <*> t <*> V)$$

Top-down parsing



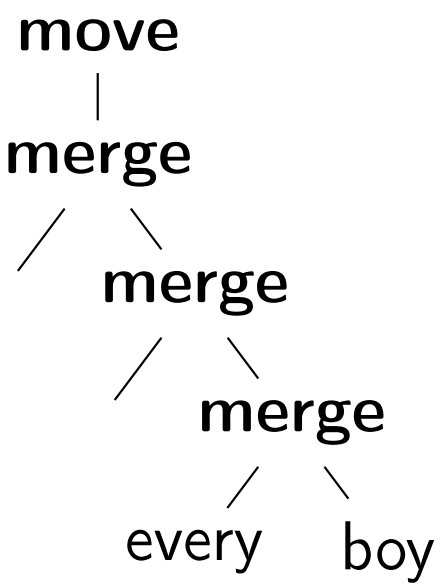
$$\lambda t, V, d, n.$$
$$(d <*> n) <*> (\circ^{\uparrow} <*> t <*> V)$$

Top-down parsing



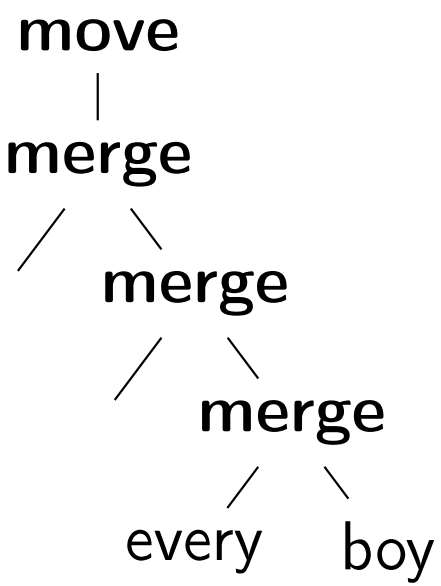
$$\lambda t, V, n.$$
$$(every^{\uparrow} <*> n) <*> (\circ^{\uparrow} <*> t <*> V)$$

Top-down parsing



$$\lambda t, V.$$
$$(every^{\uparrow} <*> boy^{\uparrow}) <*> (\circ^{\uparrow} <*> t <*> V)$$

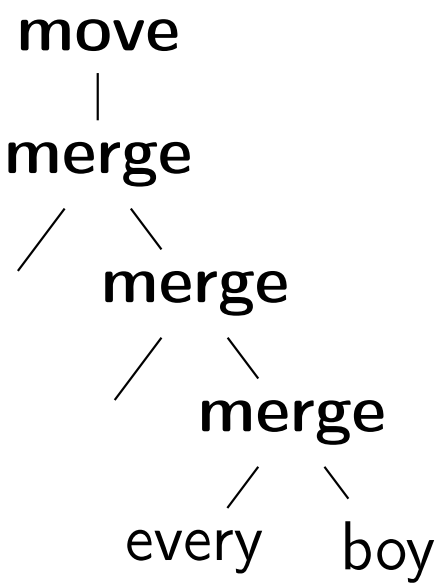
Top-down parsing



$$\lambda t, V.$$

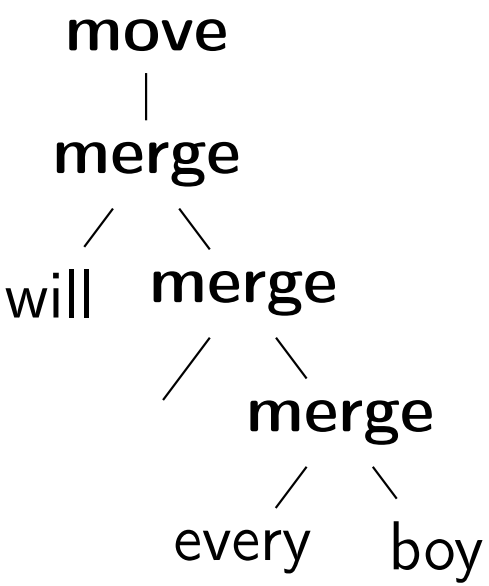
$$(\mathit{every}^{\uparrow} <*> \mathit{boy}^{\uparrow}) <*> (\circ^{\uparrow} <*> t <*> V)$$

Top-down parsing



$$\lambda t, V.$$
$$(every\ \ boy)^{\uparrow} <*> (\circ^{\uparrow} <*> t <*> V)$$

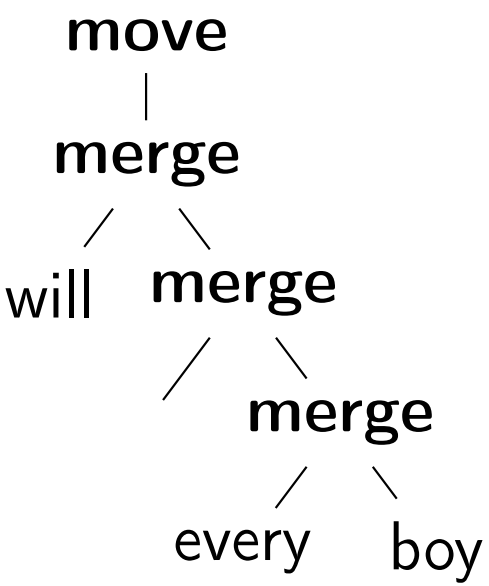
Top-down parsing



$$\lambda V.$$

$$(every\ \ boy)^{\uparrow} <*> (\circ^{\uparrow} <*> will^{\uparrow} <*> V)$$

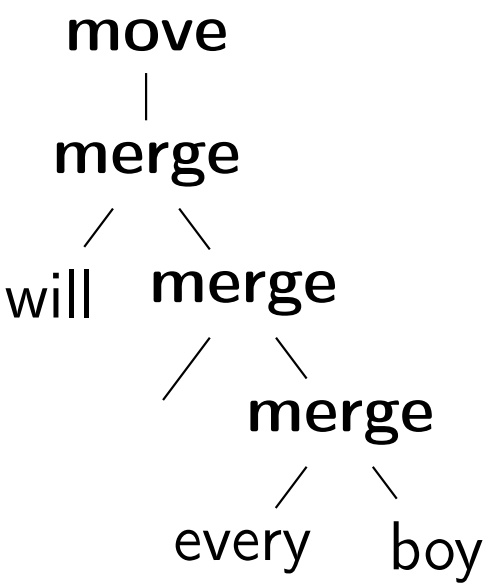
Top-down parsing



$$\lambda V.$$

$$(every\ \ boy)^{\uparrow} <*> (\circ^{\uparrow} <*> will^{\uparrow} <*> V)$$

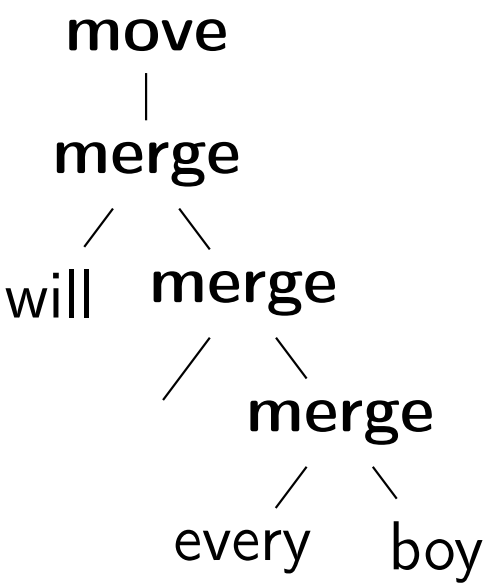
Top-down parsing



$$\lambda V.$$

$$(every\ \ boy)^{\uparrow} <*> ((\lambda f.will \circ f)^{\uparrow} <*> V)$$

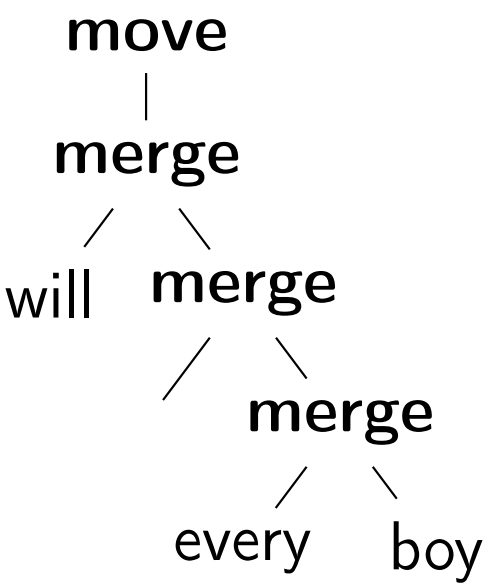
Top-down parsing



$$\lambda V.$$

$$(every\ \ boy)^{\uparrow} <*> ((\lambda f.will \circ f)^{\uparrow} <*> V)$$

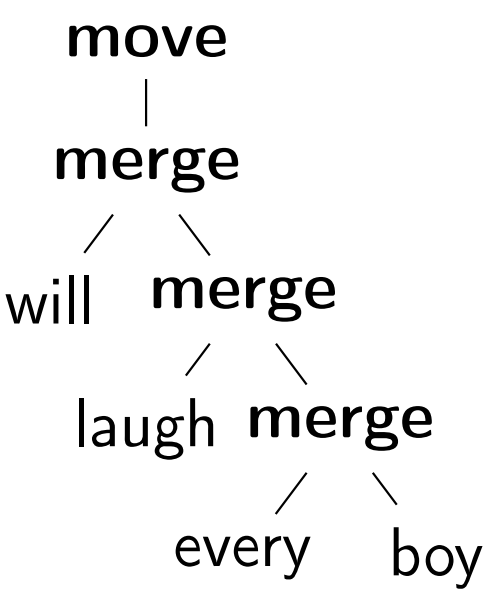
Top-down parsing



$$\lambda V.$$

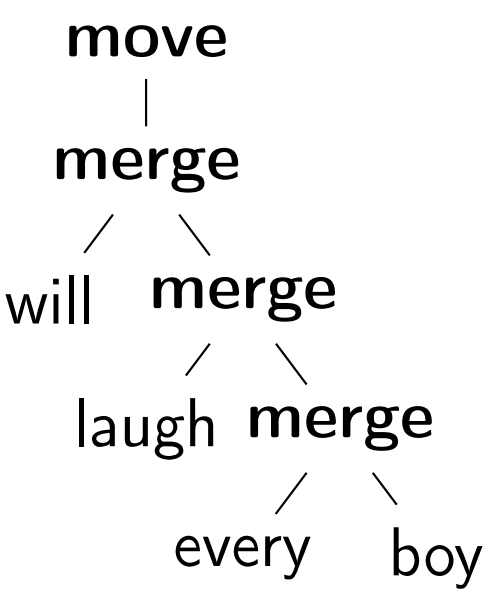
$$(\lambda P.every\ boy(\lambda x.will\ (P\ x)))^{\uparrow} <*> V$$

Top-down parsing



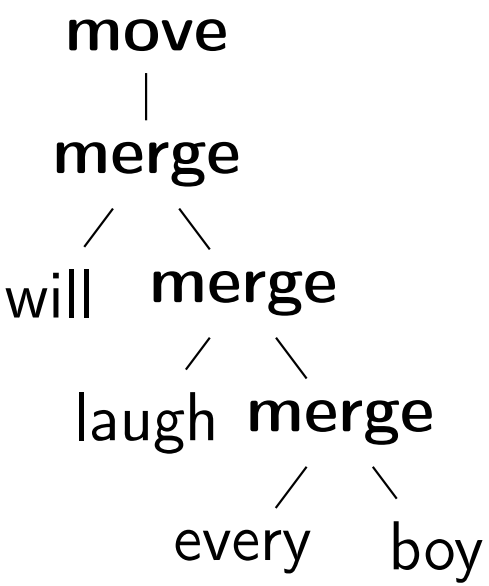
$$(\lambda P.every\ boy(\lambda x.will\ (P\ x)))^{\uparrow} <*> laugh^{\uparrow}$$

Top-down parsing



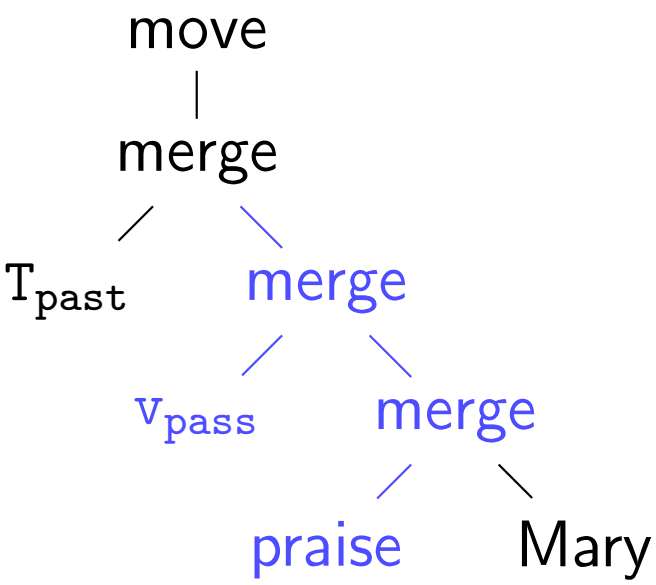
$$(\lambda P.every\ boy(\lambda x.will\ (P\ x)))^{\uparrow} <*> laugh^{\uparrow}$$

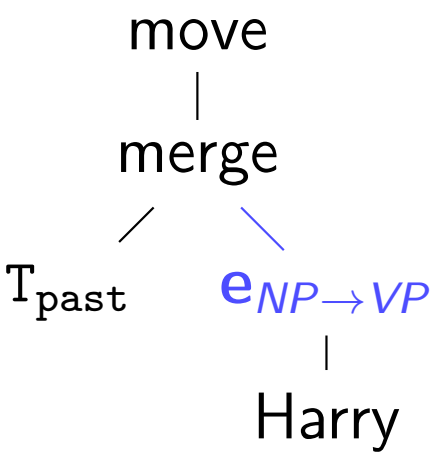
Top-down parsing



$$(every\ boy(\lambda x.will\ (laugh\ x)))^{\uparrow}$$

Ellipsis





$$[\![\,\text{move}\,]\!](\,[\![\,\text{merge}\,]\!]([\![\mathtt{T}_{\mathtt{pst}}]\!])([\![\,\mathbf{e}\,]\!]([\![\mathrm{Harry}]\!])))$$

$$\lambda m.(\mathbf{Pass} \circ \mathbf{praise})^{\uparrow} <\!*\!> \Box m$$

Plan

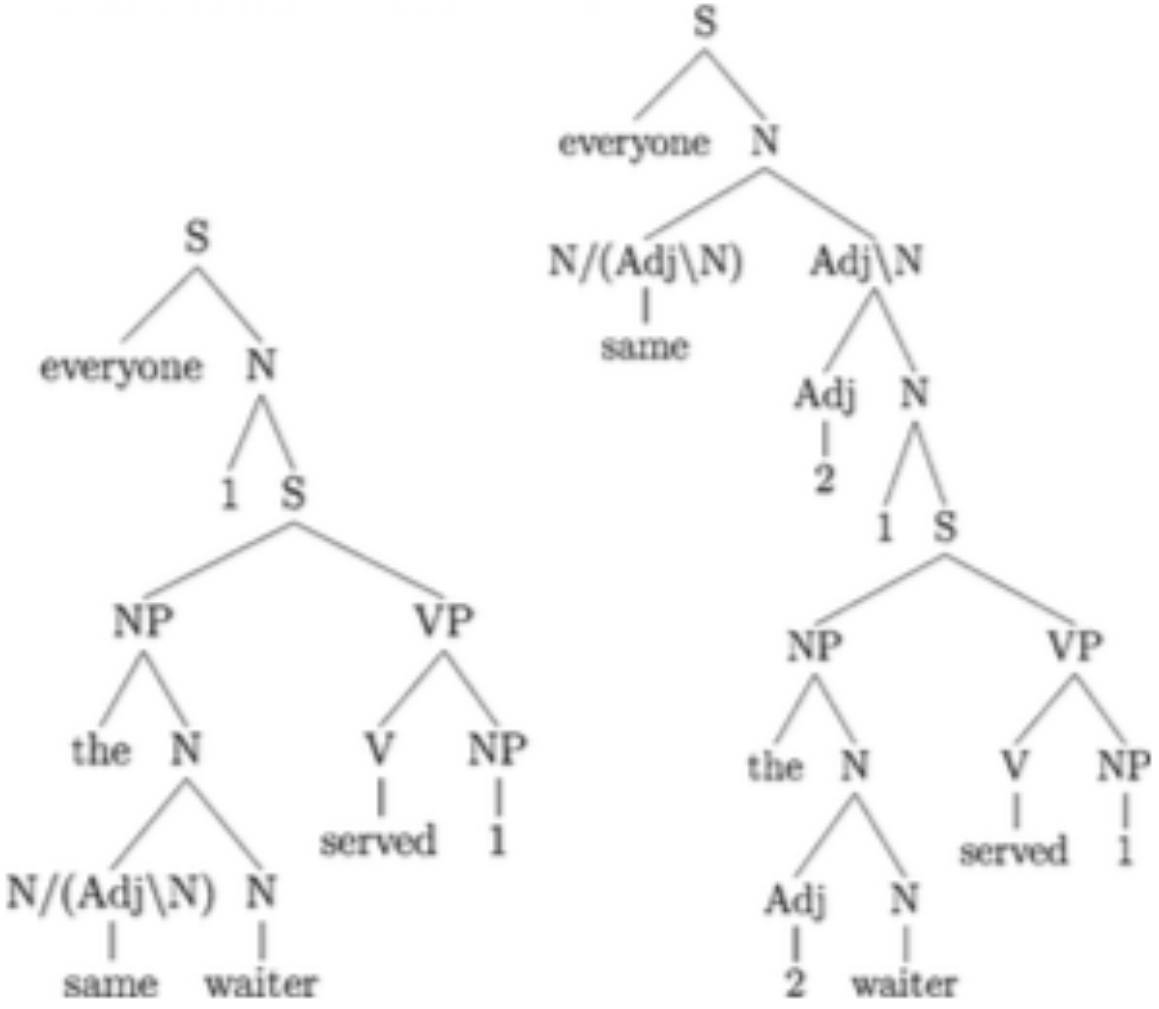
- give compositional semantics for parasitic scope
- ibid. for tucking-in movement

Main claim

Tucking-in type movement involves complex quantifier formation

Parasitic Scope

Parasitic Scope:



Before all movement

$$[\mathit{same}]_Q, [\mathit{everyone}]_x \vdash \mathit{serve}\ x\ (\mathit{the}\ (Q\ \mathit{waiter}))$$

The *types* of the stored elements are as follows:

same $(\mathbf{A}et)et$ $\qquad$ $\mathbf{A} = (et)et$

everyone $(et)t$

Before all movement

$$[\mathit{same}]_Q, [\mathit{everyone}]_x \vdash \mathit{serve}\ x\ (\mathit{the}\ (Q\ \mathit{waiter}))$$

The *types* of the stored elements are as follows:

same $(\mathbf{A}et)et$

everyone $(et)t$

$\mathbf{A} = (et)et$

What we want:

$$\vdash \mathit{everyone}\ (\mathit{same}\ (\lambda Q, x.\mathit{serve}\ x\ (\mathit{the}\ (Q\ \mathit{waiter}))))$$

Before all movement

$$[\mathit{same}]_Q, [\mathit{everyone}]_x \vdash \mathit{serve}\ x\ (\mathit{the}\ (Q\ \mathit{waiter}))$$

The *types* of the stored elements are as follows:

same $(\mathbf{A}et)et$ $\qquad$ $\mathbf{A} = (et)et$

everyone $(et)t$

What we want:

$$\vdash (\mathit{everyone} \circ \mathit{same})\ (\lambda Q, x.\mathit{serve}\ x\ (\mathit{the}\ (Q\ \mathit{waiter})))$$

Parasitic Storage (Sketch)

$$\dfrac{\dfrac{[\mathit{same}]_Q, [\mathit{everyone}]_x \vdash \mathit{serve}\ x\ (\mathit{the}\ (Q\ \mathit{waiter}))}{[\mathit{everyone} \circ \mathit{same}]_{Q;x} \vdash \mathit{serve}\ x\ (\mathit{the}\ (Q\ \mathit{waiter}))}}{\vdash (\mathit{everyone} \circ \mathit{same})\ (\lambda Q, x.\mathit{serve}\ x\ (\mathit{the}\ (Q\ \mathit{waiter})))}$$

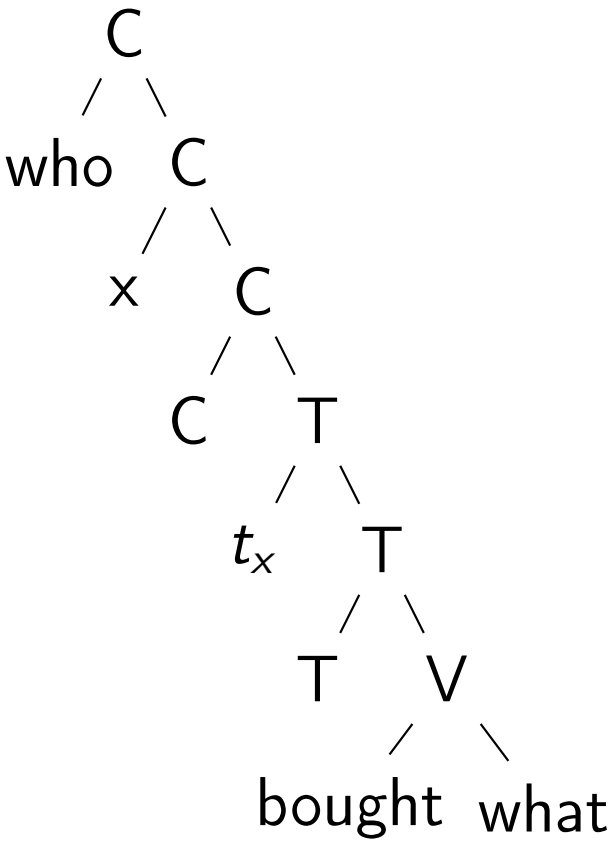
Parasitic Storage

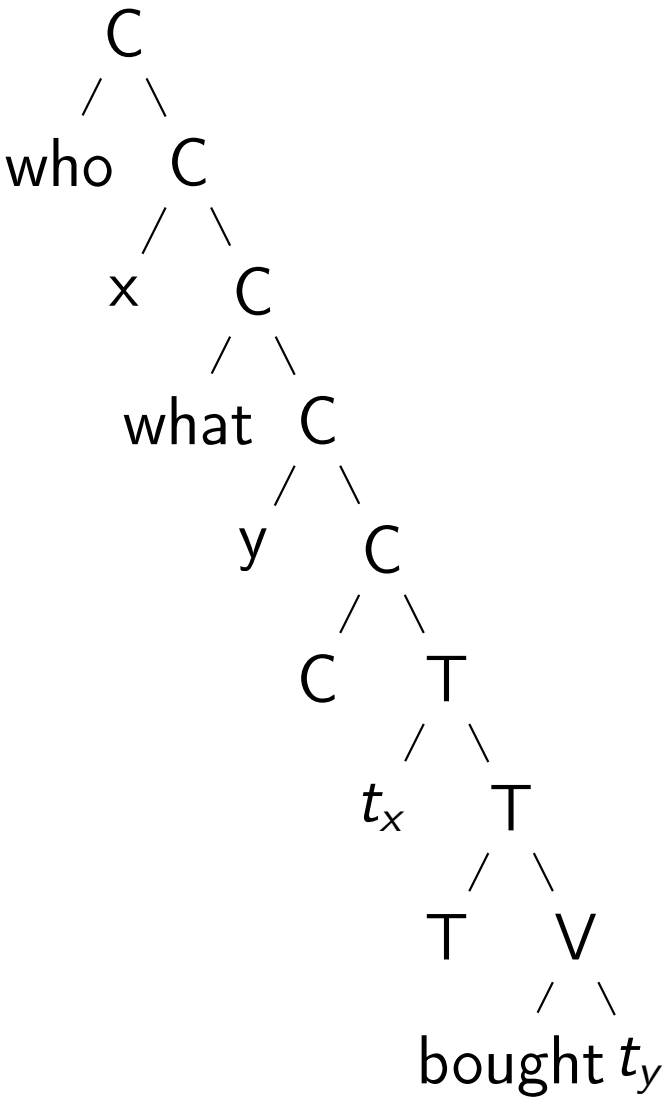
$$\frac{[\mathit{same}]_Q, [\mathit{everyone}]_x \vdash \mathit{serve}\ x\ (\mathit{the}\ (Q\ \mathit{waiter}))}{[\Pi\ \mathit{everyone}\ \mathit{same}]_z \vdash \text{let}\ (x, Q)\ \text{be}\ z\ \text{in}\ \mathit{serve}\ x\ (\mathit{the}\ (Q\ \mathit{waiter}))}$$

$$\frac{\Gamma, [M : (\alpha\beta)\gamma]_x, [N : (\delta\alpha\eta)\alpha\beta]_y, \Delta \vdash O : \zeta}{\Gamma, [\Pi\ M\ N : ((\delta \otimes \alpha)\eta)\gamma]_z, \Delta \vdash \text{let}\ (x, y)\ \text{be}\ z\ \text{in}\ O : \zeta}\ \text{fuse}$$

where $\Pi\ M\ N := M \circ N \circ \texttt{flip} \circ \texttt{curry} = \lambda R.M\ (N\ (\lambda y, x.R\ (x, y)))$

Tucking-in Movement





Before all movement

$$[who]_x, [what]_y \vdash bought\ y\ x$$

Before all movement

$$[who]_x, [what]_y \vdash bought\ y\ x$$

What we want:

$$\vdash who\ (\lambda x.what\ (\lambda y.bought\ y\ x))$$

Before all movement

$$[who]_x, [what]_y \vdash bought\ y\ x$$

What we want:

$$\vdash (who \circ (\lambda R, x.what\ (\lambda y.R\ y\ x)))\ bought$$

Before all movement

$$[who]_x, [what]_y \vdash bought\ y\ x$$

What we want:

$$\vdash (who \circ (\mathbb{L}\ what))\ bought$$

$$\begin{aligned}\mathbb{L} &:= \mathbb{B}(\mathbb{CBC})\mathbb{B} \\ &= \lambda D, R, x.D(\lambda y.Ryx)\end{aligned}$$

About $\mathbb{L}$

GQs as *arity reducers* (Keenan, 2016)

A DP denotes a function of type $\forall n.(e^{n+1}t)e^n t$

For $D : (et)t$

$$\mathbb{L}^n\ D : (e^{n+1}t)e^n t$$
$$= \lambda R, x_1, \ldots, x_n.D\ (\lambda y.R\ y\ x_1\ \ldots\ x_n)$$

Tucked Storage (Sketch)

$$\dfrac{\dfrac{[who]_x, [what]_y \vdash bought\ y\ x}{[who \circ (\mathbb{L}\ what)]_{x;y} \vdash bought\ y\ x}}{\vdash (who \circ (\mathbb{L}\ what))\ (\lambda x, y.bought\ y\ x)}$$

Tucked Storage

$$\frac{[\mathit{who}]_x, [\mathit{what}]_y \vdash \mathit{bought}\ y\ x}{[\Pi\ \mathit{who}\ (\mathbb{L}\ \mathit{what})]_z \vdash \text{let } (x, y) \text{ be } z \text{ in } \mathit{bought}\ y\ x}$$

$$\frac{\Gamma, [M : (et)t]_x, [N : (et)t]_y, \Delta \vdash O : \zeta}{\Gamma, [\Pi\ M\ (\mathbb{L}\ N) : ((e \otimes e)t)t]_z, \Delta \vdash \text{let } (x, y) \text{ be } z \text{ in } O : \zeta}\ \text{tuck}$$

where $\Pi\ M\ N := M \circ N \circ \texttt{flip} \circ \texttt{curry} = \lambda R.M\ (N\ (\lambda y, x.R\ (x, y)))$

Parasitic vs Tucked Storage

$$\frac{[\mathit{everyone}]_x, [\mathit{same}]_Q \vdash \mathit{serve}\ x\ (\mathit{the}\ (Q\ \mathit{waiter}))}{[\Pi\ \mathit{everyone}\ \mathit{same}]_z \vdash \text{let}\ (x, Q)\ \text{be}\ z\ \text{in}\ \mathit{serve}\ x\ (\mathit{the}\ (Q\ \mathit{waiter}))}$$

$$\frac{[\mathit{who}]_x, [\mathit{what}]_y \vdash \mathit{bought}\ y\ x}{[\Pi\ \mathit{who}\ (\mathbb{L}\ \mathit{what})]_z \vdash \text{let}\ (x, y)\ \text{be}\ z\ \text{in}\ \mathit{bought}\ y\ x}$$

Parasitic vs Tucked Storage

$$\frac{[M]_x, [N]_y \vdash O}{[\Pi\ M\ N]_z \vdash \text{let } (x, y) \text{ be } z \text{ in } O}$$

$$\frac{[M]_x, [N]_y \vdash O}{[\Pi\ M\ (\mathbb{L}\ N)]_z \vdash \text{let } (x, y) \text{ be } z \text{ in } O}$$

Parasitic vs Tucked Storage

$$\frac{[M]_x, [N]_y \vdash O}{[\Pi\ M\ (\mathbb{L}^n\ N)]_z \vdash \text{let } (x, y) \text{ be } z \text{ in } O}$$

Conclusion

LF-interpretation is in fact directly compositional

- This gives us access to a wealth of results and tools
  - generative capacity
  - generation
  - delimited continuations
- gives new questions to explore
  - semantic bootstrapping?!
  - conditioning parsing on meaning?!
  - connex to cogsci via probabilistic programming?!
- and addresses old ones
  - LF? Ellipsis? ACD? ...