Direct Compositionality

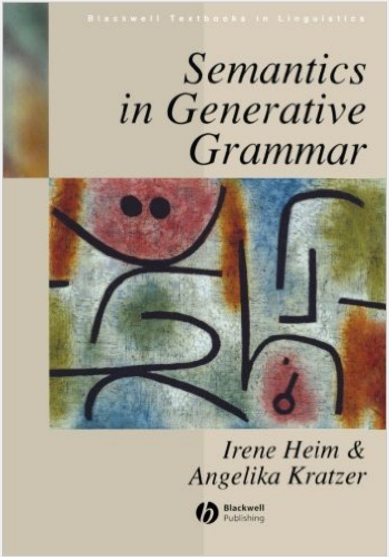
Greg Kobele

Universität Leipzig

Winter Semester, 2020

H&K and Compositionality

Semantics in Generative Grammar



- Binary branching nodes
  $$\left\llbracket \begin{array}{ccc} & \bullet & \\ / & & \backslash \\ \alpha & & \beta \end{array} \right\rrbracket^g = \llbracket\alpha\rrbracket^g \oplus \llbracket\beta\rrbracket^g$$
- Unary branching nodes
  $$\left\llbracket \begin{array}{c} \bullet \\ | \\ \alpha \end{array} \right\rrbracket^g = \llbracket\alpha\rrbracket^g$$
- Binding
  $$\left\llbracket \begin{array}{ccc} & \bullet & \\ / & & \backslash \\ i & & \alpha \end{array} \right\rrbracket^g = \lambda x.\llbracket\alpha\rrbracket^{g[i:=x]}$$
- Traces
  $\llbracket t_i \rrbracket^g = g(i)$

Most expressions don't have any meaning

$$[\![\, [_{V}\ [_{V}\ \text{praise}]\ [_{D}\ [_{D}\ \text{every}]\ [_{N}\ \text{boy}]]] \,]\!]^{g} = [\![\textit{praise}]\!]^{g} \oplus [\![\, [_{D}\ [_{D}\ \text{every}]\ [_{N}\ \text{boy}]] \,]\!]^{g}$$

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

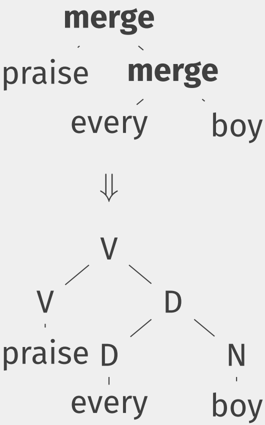
$[\![\textit{every}]\!]^{g} \oplus [\![\textit{boy}]\!]^{g} : (et)t$ $\qquad$ $[\![\textit{praise}]\!]^{g} : eet$

these cannot be combined!

FA $\alpha\beta \to \alpha \to \beta$ $\qquad$ 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 $\mathbf{every}(\mathbf{boy})(\lambda x. \ldots$
and a property part $\mathbf{praise}(x)$

Let's write instead:

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

Notation and Operations

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

the general case:

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

The entire point

is to ignore what is stored

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

Working with Storage

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

BUILDING *PRAISE EVERY BOY*

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

Building *praise every boy*

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

We want to 'insert a trace'

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

Building *praise every boy*

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

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

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

retrieval

$$\frac{\Gamma, [M_i]_{x_i}, \Delta \vdash N}{\Gamma, \Delta \vdash M_i \oplus (\lambda x_i.N)}\,[\oplus]_i$$

Taking things out of storage

$$\dfrac{\dfrac{\textbf{seem}}{\vdash \textbf{seem}}\uparrow \quad \dfrac{\dfrac{\textbf{Pass}}{\vdash \textbf{Pass}}\uparrow \quad \dfrac{\vdots}{[\textbf{every}(\textbf{boy})]_x \vdash \textbf{praise}(x)}}{[\textbf{every}(\textbf{boy})]_x \vdash \textbf{Pass}(\textbf{praise}(x))} <*>}{\dfrac{[\textbf{every}(\textbf{boy})]_x \vdash \textbf{seem}(\textbf{Pass}(\textbf{praise}(x)))}{\vdash \textbf{every}(\textbf{boy})(\lambda x.\textbf{seem}(\textbf{Pass}(\textbf{praise}(x))))} [\text{FA}]_1} <*>$$

retrieval

$$\frac{\Gamma, [M_i]_{x_i}, \Delta \vdash N}{\Gamma, \Delta \vdash M_i \oplus (\lambda x_i.N)} [\oplus]_i$$

Manipulating Stores

pure

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

apply

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

retrieve

$$\frac{\Gamma, [M_i]_{x_i}, \Delta \vdash N}{\Gamma, \Delta \vdash M_i \oplus (\lambda x_i.N)}\ [\oplus]_i$$

store

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

More notation

idiom brackets

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

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

application

Forward $f \triangleright a := f\ a$

Backward $a \triangleleft f := f\ a$

Unpacking the notation

Recall that

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

means

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

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

Minimalist semantics

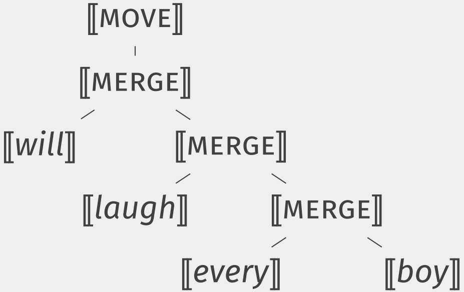
$$[\![\textbf{merge}]\!] \mapsto \lambda m, n.(\!| m \oplus n |\!)$$
$$[\![\textbf{merge}]\!] \mapsto \lambda m, n.(\!| m \oplus \Box n |\!)$$

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

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

for $\oplus \in \{\rhd, \lhd\}$

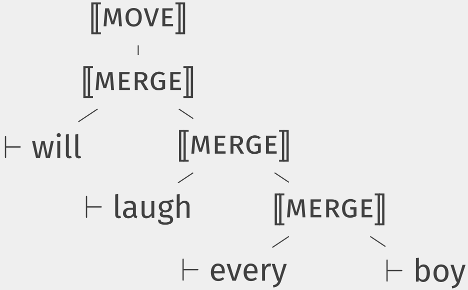
EXAMPLE



Example

- $[\![\text{MOVE}]\!]$
  - $[\![\text{MERGE}]\!]$
    - $\mathcal{I}(\textit{will})^{\uparrow}$
    - $[\![\text{MERGE}]\!]$
      - $\mathcal{I}(\textit{laugh})^{\uparrow}$
      - $[\![\text{MERGE}]\!]$
        - $\mathcal{I}(\textit{every})^{\uparrow}$
        - $\mathcal{I}(\textit{boy})^{\uparrow}$

Example



Example

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

- $[\![\textsc{merge}]\!]$
  - $\vdash$ will
  - $[\![\textsc{merge}]\!]$
    - $\vdash$ laugh
    - $\lambda m, n.(\!| m \rhd n |\!)$
      - $\vdash$ every
      - $\vdash$ boy

Example

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

$[\![\textsc{merge}]\!]$

$\vdash$ will $\quad$ $[\![\textsc{merge}]\!]$

$\vdash$ laugh $\quad$ $\vdash$ every boy

Example

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

$[\![\textsc{merge}]\!]$

$\vdash$ will $\quad \lambda m, n.(\!| m \rhd \Box n |\!)$

$\vdash$ laugh $\quad \vdash$ every boy

Example

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

$$[\![\textsc{merge}]\!]$$

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

Example

$$\begin{array}{c}
[\![\textsc{move}]\!] \\
\lambda m, n.(\!|m \rhd n|\!) \\
\vdash \text{will} \qquad [\text{every boy}]_x \vdash \text{laugh } x
\end{array}$$

Example

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

Example

$$\lambda m.[\triangleright]_1 m$$

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

Example

$\vdash$ every boy (λx.will (laugh x))

Derivations

Plan

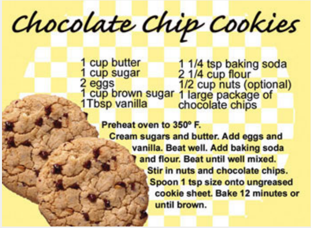
- Explain what derivations are
- Show relation between derivations and more familiar derived structures

Main claim

Syntactic structures are and *always have been* derivations

Derivations are recipes

- lexical items are ingredients
- merge and move instead of bake, beat, stir ...



Derivations are structured

Order is important

- Some things must happen before others
- Sometimes, it doesn't matter

- merge det and noun
- *before* you merge the verb

- cream sugar and butter
- *before* you add the flour

Representing derivations

Representing derivations

1. `select` *every*

every

Representing derivations

1. `select` *every*
2. `select` *boy*

every boy

Representing derivations

1. `select` *every*
2. `select` *boy*
3. `merge` *1* and *2*
   $[_{DP}\ \text{every}\ [_{NP}\ \text{boy}\ ]]$

MERGE

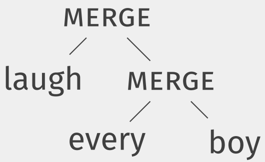
every boy

Representing derivations

1. `select` *every*
2. `select` *boy*
3. `merge` *1* and *2*
   $[_{DP}\ \text{every}\ [_{NP}\ \text{boy}\ ]]$
4. `select` *laugh*

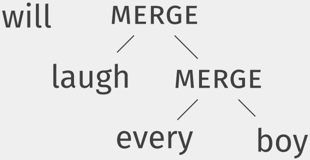
Representing derivations

1. `select` *every*
2. `select` *boy*
3. `merge` *1* and *2*
   [$_{DP}$ every [$_{NP}$ boy]]
4. `select` *laugh*
5. `merge` *4* and *3*
   [$_{VP}$ laugh [$_{DP}$ every boy]]



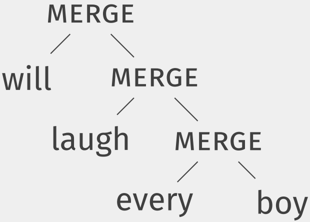
Representing derivations

1. `select` *every*
2. `select` *boy*
3. `merge` *1* and *2*
   $[_{DP}$ every $[_{NP}$ boy $]]$
4. `select` *laugh*
5. `merge` *4* and *3*
   $[_{VP}$ laugh $[_{DP}$ every boy $]]$
6. `select` *will*



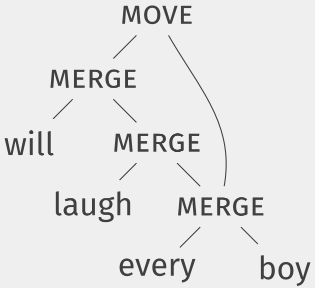
Representing derivations

1. `select` *every*
2. `select` *boy*
3. `merge` *1* and *2*
   $[_{DP}$ every $[_{NP}$ boy $]]$
4. `select` *laugh*
5. `merge` *4* and *3*
   $[_{VP}$ laugh $[_{DP}$ every boy $]]$
6. `select` *will*
7. `merge` *6* and *5*
   $[_{IP}$ will $[_{VP}$ laugh $[_{DP}$ every boy $]]]$

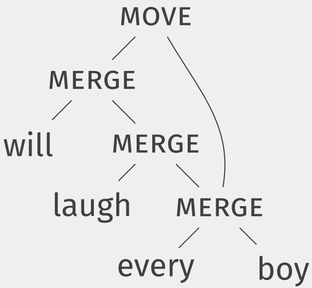


Representing derivations

1. `select` *every*
2. `select` *boy*
3. `merge` *1* and *2*
   [$_{DP}$ every [$_{NP}$ boy]]
4. `select` *laugh*
5. `merge` 4 and *3*
   [$_{VP}$ laugh [$_{DP}$ every boy]]
6. `select` *will*
7. `merge` *6* and *5*
   [$_{IP}$ will [$_{VP}$ laugh [$_{DP}$ every boy]]]
8. `move` *every boy*
   [$_{IP}$[$_{DP}$ every boy][$_{I'}$ will [$_{VP}$ laugh *t*]]]



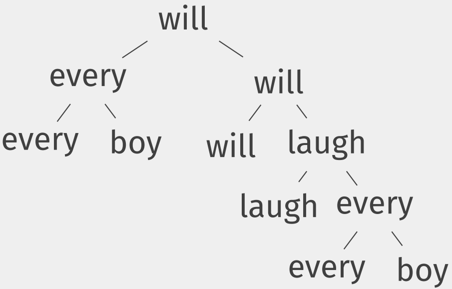
The structure of derivations



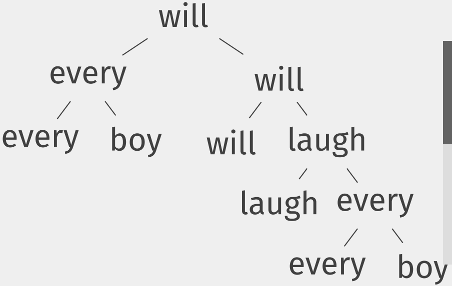
x dominates y: *x* was built using *y*

x c-commands y: *x* 's sister was built using *y*

Structure in Minimalism

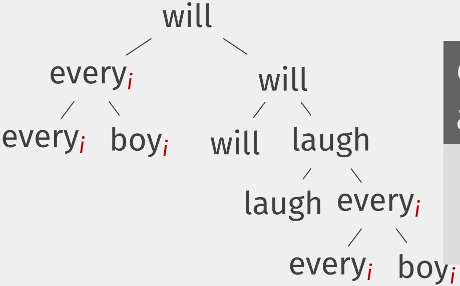


Structure in Minimalism



occurrences of *every boy* are "non-distinct"

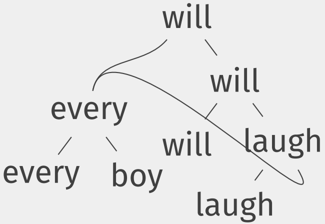
Structure in Minimalism



occurrences of *every boy* are "non-distinct"

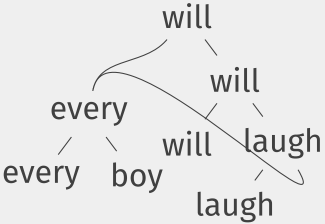
- coindexation

Structure in Minimalism



occurrences of *every boy* are "non-distinct"

- coindexation
- multiple dominance



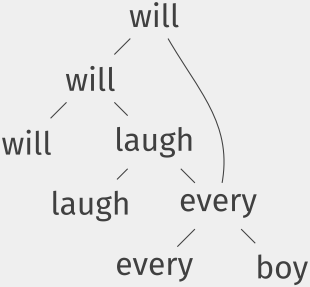
occurrences of *every boy* are "non-distinct"

- coindexation
- multiple dominance

Antisymmetry

Order not meaningful

Structure in Minimalism



occurrences of *every boy* are "non-distinct"

- coindexation
- multiple dominance

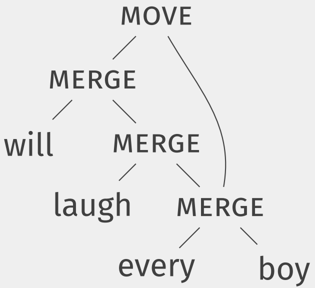
Antisymmetry

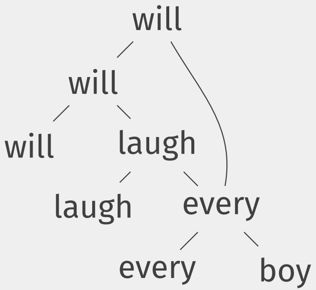
Order not meaningful

Derivations of Derived Structures

every boy will laugh

- has the structure on the right
- constructed via the process on the left

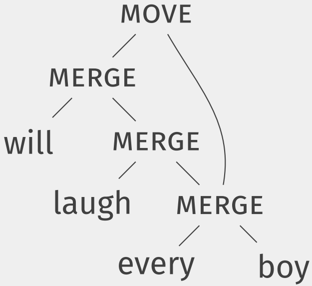


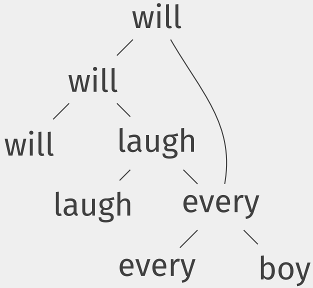


Derivations of Derived Structures

The derivation is building a copy of itself

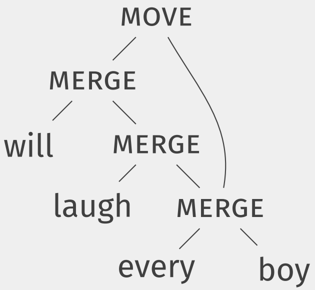
Derived structure is a reification of
the structure of the derivational process

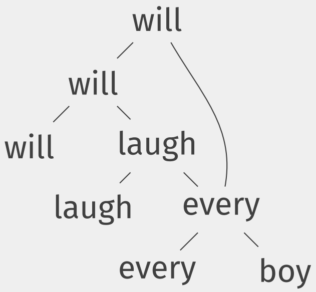




Derivations of Derived Structures

We have been writing derivation trees all along





The derivational perspective

Structure = derivation

the derivational process structures expressions
in just the way we want

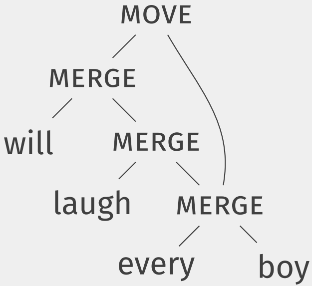
Practical consequences

no post-facto alteration of structure
build it the way you want it

Conceptual benefit

two structures are identical
when they describe the same process

The Determinacy of Movement



Attract Closest

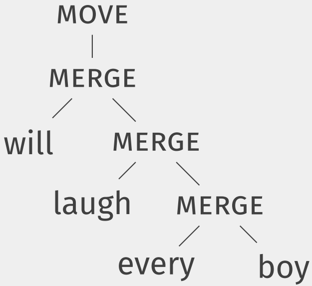
Minimal Link

Shortest Move

SMC

can only be 1 thing moving for a particular reason at any time

The Determinacy of Movement



Attract Closest

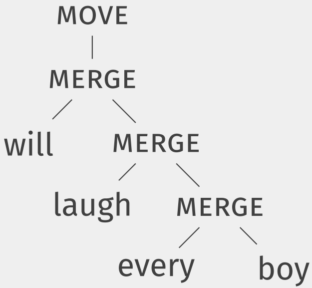
Minimal Link

Shortest Move

SMC

can only be 1 thing moving for a particular reason at any time

The Determinacy of Movement



$\text{MERGE}(\alpha, \beta) = \{\alpha, \beta\}$

$\text{MOVE}(\alpha) = \text{MERGE}(\alpha, \alpha) = \{\alpha\}$

$\{\{\text{will}, \{\text{laugh}, \{\text{every}, \text{boy}\}\}\}\}$

- No tampering
- No indices
- No lexical (sub-)arrays

A familiar picture

> *Syntactic structure is no more than the trace of the algorithm which delivers the interpretation*
>
> (Steedman, 2000)