Sharpening the empirical claims of generative syntax through formalization

Tim Hunter

University of Minnesota, Twin Cities

ESSLLI, August 2015

Sharpening the empirical claims of generative syntax through formalization

Tim Hunter — ESSLLI, August 2015

Part 1

Grammars and Cognitive Hypotheses

Outline

Outline

Claims made by grammars

What are grammars used for?

- "Mostly" for accounting for acceptability judgements
- But there are other ways a grammar can figure in claims about cognition

Claims made by grammars

What are grammars used for?

- "Mostly" for accounting for acceptability judgements
- But there are other ways a grammar can figure in claims about cognition

Often tempting to draw a distinction between "linguistic evidence" (where grammar lives) and "experimental evidence" (where cognition lives)

- One need not make this distinction
- We will proceed without it, i.e. it's all linguistic (and/or all experimental)

Claims made by grammars

There's a "boring" sense in which every syntax paper makes a cognitive claim, i.e. a claim testable via acceptability facts.

Claims made by grammars

There's a "boring" sense in which every syntax paper makes a cognitive claim, i.e. a claim testable via acceptability facts.

- For one thing, this is not a cop-out!
  - Why does it seem like a cop-out?
  - Lingering externalism/Platonism?
  - Perhaps partly because it's just relatively rare to see anything being tested by other measures

Claims made by grammars

There's a "boring" sense in which every syntax paper makes a cognitive claim, i.e. a claim testable via acceptability facts.

- For one thing, this is not a cop-out!
  - Why does it seem like a cop-out?
  - Lingering externalism/Platonism?
  - Perhaps partly because it's just relatively rare to see anything being tested by other measures
- For another, we can incorporate grammars into claims that are testable by other measures.
  - This is the main point of the course!
  - The claims/predictions will depend on internal properties of grammars, not just what they say is good and what they say is bad
  - And we'll do it without seeing grammatical derivations as real-time operations

Claims made by grammars

If we accept — as I do — . . . that the rules of grammar enter into the processing mechanisms, then evidence concerning production, recognition, recall, and language use in general can be expected (in principle) to have bearing on the investigation of rules of grammar, on what is sometimes called "grammatical competence" or "knowledge of language".

(Chomsky 1980: pp.200-201)

[S]ince a competence theory must be incorporated in a performance model, evidence about the actual organization of behavior may prove crucial to advancing the theory of underlying competence.

(Chomsky 1980: p.226)

Claims made by grammars

> *If we accept — as I do — . . . that the rules of grammar enter into the processing mechanisms, then evidence concerning production, recognition, recall, and language use in general can be expected (in principle) to have bearing on the investigation of rules of grammar, on what is sometimes called "grammatical competence" or "knowledge of language".*
>
> *(Chomsky 1980: pp.200-201)*

> *[S]ince a competence theory must be incorporated in a performance model, evidence about the actual organization of behavior may prove crucial to advancing the theory of underlying competence.*
>
> *(Chomsky 1980: p.226)*

Evidence about X can only advance Y if Y makes claims about X!

Preview

What we will do:

- Put together a chain of linking hypotheses that bring “experimental evidence” to bear on “grammar questions”
  - e.g. reading times, acquisition patterns
  - e.g. move as distinct operation from merge vs. unified with merge
- Illustrate with some toy examples

Preview

What we will do:

- Put together a chain of linking hypotheses that bring "experimental evidence" to bear on "grammar questions"
  - e.g. reading times, acquisition patterns
  - e.g. move as distinct operation from merge vs. unified with merge
- Illustrate with some toy examples

What we will not do:

- Engage with state-of-the-art findings in the sentence processing literature
- End up with claims that one particular set of derivational operations is empirically better than another

Teasers

We'll take pairs of equivalent grammars that differ only in the move/re-merge dimension.

- They will make different predictions about sentence comprehension difficulty.
- They will make different predictions about what a learner will conclude from a common input corpus.

Teasers

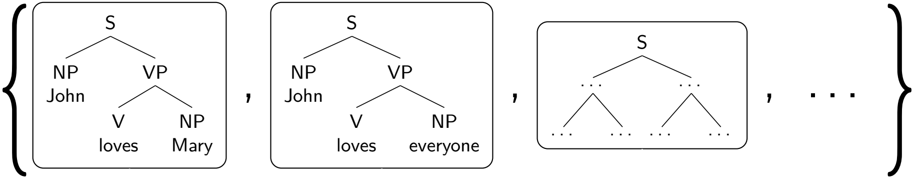
We'll take pairs of equivalent grammars that differ only in the move/re-merge dimension.

- They will make different predictions about sentence comprehension difficulty.
- They will make different predictions about what a learner will conclude from a common input corpus.

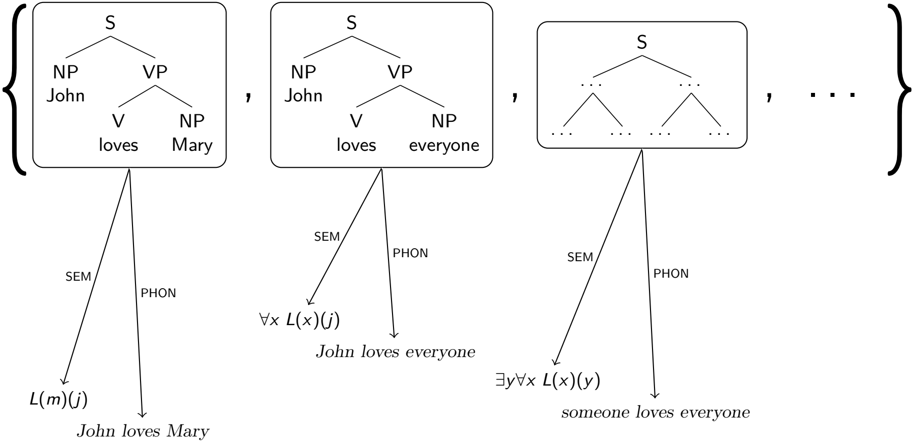
The issues become "distant but empirical questions". That's all we're aiming for, for now.

Outline

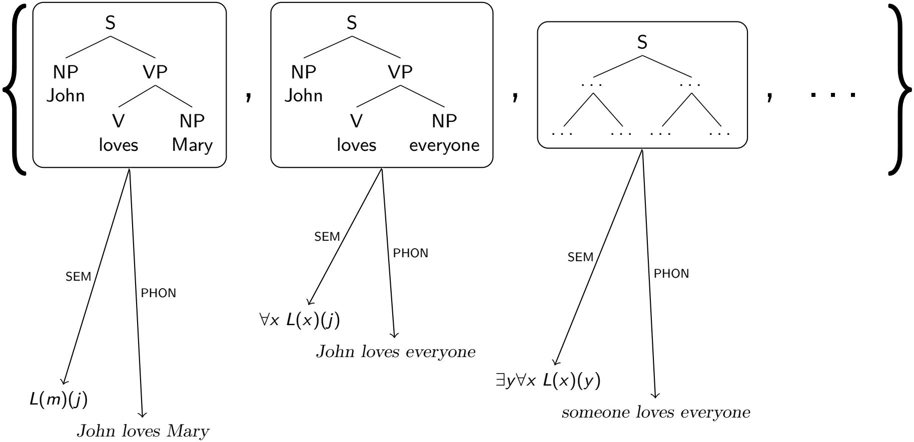
Interpretation functions



Interpretation functions



Interpretation functions



Caveats:

- Maybe we're interested in the finite specification of the set
- Maybe there's no clear line between observable and not
- Maybe some evidence is based on relativities among interpretations

Telling grammars apart

So, what if we have two different grammars — systems that define different sets of objects — that we can't tell apart via the sound and meaning interpretations?

(Perhaps because they're provably equivalent, or perhaps because the evidence just happens to be unavailable.)

Telling grammars apart

So, what if we have two different grammars — systems that define different sets of objects — that we can't tell apart via the sound and meaning interpretations?

(Perhaps because they're provably equivalent, or perhaps because the evidence just happens to be unavailable.)

- Option 1: Conclude that the differences are irrelevant to us (or "they're not actually different").
- Option 2: Make the differences matter . . . somehow . . .

What are syntactic representations for?

Morrill (1994) in favour of Option 1:

> *The construal of a language as a collection of signs [sound-meaning pairs] presents as an investigative task the characterisation of this collection. This is usually taken to mean the specification of a set of "structural descriptions" (or: "syntactic structures"). Observe however that on our understanding a sign is an association of prosodic [phonological] and semantic properties. It is these properties that can be observed and that are to be modelled. There appears to be no observation which bears directly on syntactic as opposed to prosodic and/or semantic properties, and this implies an asymmetry in the status of these levels.* ***A structural description is only significant insofar as it is understood as predicting prosodic and semantic properties (e.g. in interpreting the yield of a tree as word order). Attribution of syntactic (or prosodic or semantic) structure does not of itself predict anything.***

What are syntactic representations for?

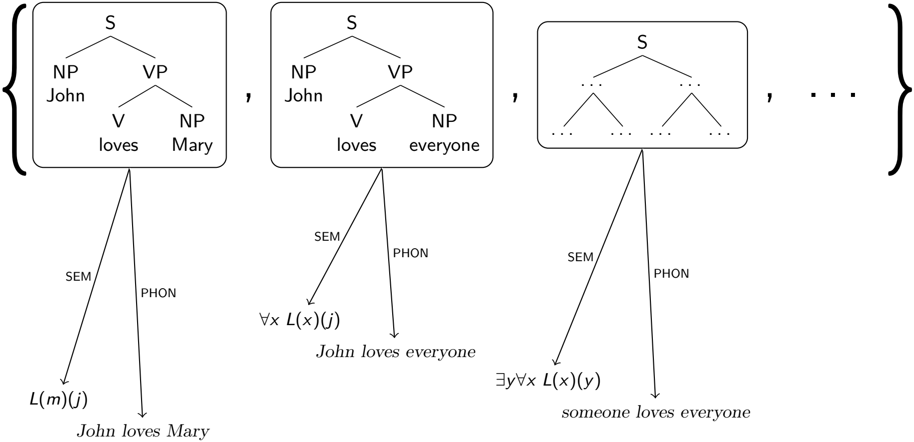
Morrill (1994) in favour of Option 1:

> *The construal of a language as a collection of signs [sound-meaning pairs] presents as an investigative task the characterisation of this collection. This is usually taken to mean the specification of a set of "structural descriptions" (or: "syntactic structures"). Observe however that on our understanding a sign is an association of prosodic [phonological] and semantic properties. It is these properties that can be observed and that are to be modelled. There appears to be no observation which bears directly on syntactic as opposed to prosodic and/or semantic properties, and this implies an asymmetry in the status of these levels.* ***A structural description is only significant insofar as it is understood as predicting prosodic and semantic properties (e.g. in interpreting the yield of a tree as word order). Attribution of syntactic (or prosodic or semantic) structure does not of itself predict anything.***

Where might we depart from this (to pursue Option 2)?

- Object that syntactic structure **does** matter "of itself"
- Object that prosodic and semantic properties are **not** the only ones we can observe

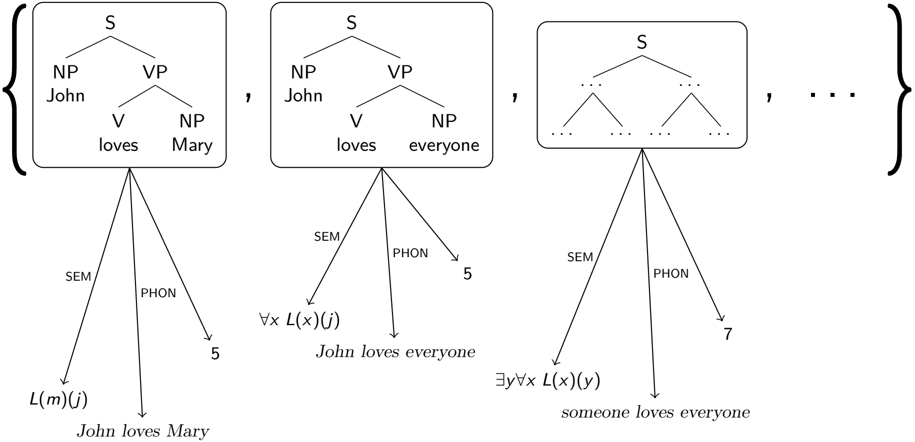
Interpretation functions



Caveats:

- Maybe we're interested in the finite specification of the set
- Maybe there's no clear line between observable and not
- Maybe some evidence is based on relativities among interpretations

Interpretation functions



Caveats:

- Maybe we're interested in the finite specification of the set
- Maybe there's no clear line between observable and not
- Maybe some evidence is based on relativities among interpretations

Interpretation functions for "complexity"

What are some other interpretation functions?

- number of nodes

Interpretation functions for “complexity”

What are some other interpretation functions?

- number of nodes
- ratio of total nodes to terminal nodes (Miller and Chomsky 1963)

Ratio of total nodes to terminal nodes

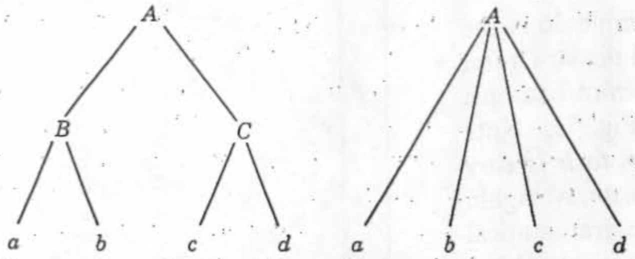


Fig. 8. Illustrating a measure of structural complexity. $N(Q)$ for the P-marker (a) is 7/4; for (b), $N(Q) = 5/4$.

(Miller and Chomsky 1963)

Ratio of total nodes to terminal nodes

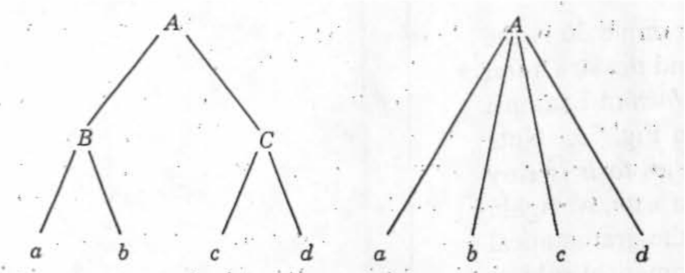


Fig. 8. Illustrating a measure of structural complexity. $N(Q)$ for the P-marker (a) is 7/4; for (b), $N(Q) = 5/4$.

Won't distinguish center-embedding from left- and right-embedding

(1) The mouse [the cat [the dog bit] chased] died. (center)

(2) The dog bit the cat [which chased the mouse [which died]]. (right)

(3) [[the dog] 's owner] 's friend (left)

(Miller and Chomsky 1963)

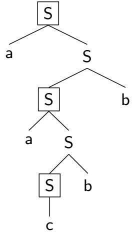
Interpretation functions for “complexity”

What are some other interpretation functions?

- number of nodes
- ratio of total nodes to terminal nodes (Miller and Chomsky 1963)
- degree of self-embedding (Miller and Chomsky 1963)

Degree of (centre-)self-embedding

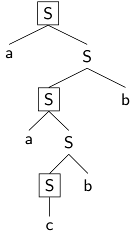
A tree's degree of self-embedding is m iff:
"there is ... a continuous path passing through $m+1$ nodes $N_0, \ldots, N_m$, each with the same label, where each N_i ($i \geq 1$) is fully self-embedded (with something to the left and something to the right) in the subtree dominated by N_{i-1}"

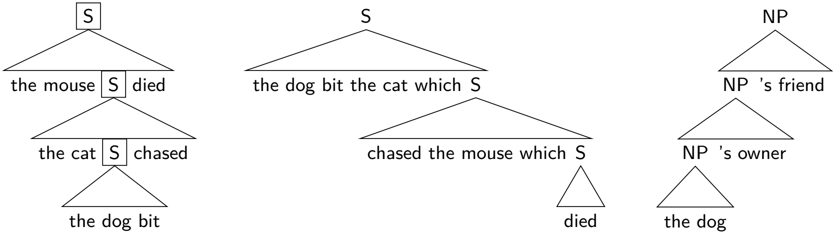


(Miller and Chomsky 1963)

Degree of (centre-)self-embedding

A tree's degree of self-embedding is m iff:
"there is ... a continuous path passing through $m+1$ nodes $N_0, \ldots, N_m$, each with the same label, where each N_i ($i \geq 1$) is fully self-embedded (with something to the left and something to the right) in the subtree dominated by N_{i-1}"





(Miller and Chomsky 1963)

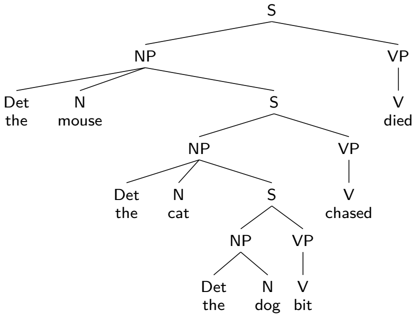
Interpretation functions for “complexity”

What are some other interpretation functions?

- number of nodes
- ratio of total nodes to terminal nodes (Miller and Chomsky 1963)
- degree of self-embedding (Miller and Chomsky 1963)
- “depth” of memory required by a top-down parser (Yngve 1960)

Yngve's depth

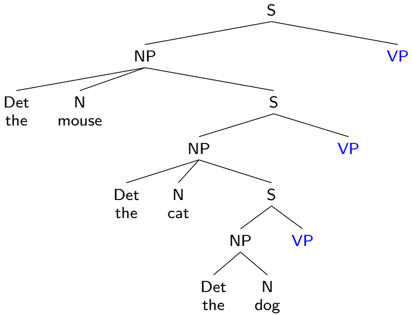
Number of constituents expected but not yet started:



(Yngve 1960)

Yngve's depth

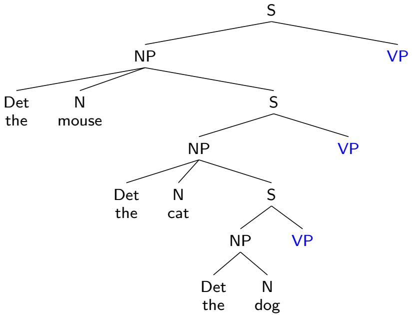
Number of constituents expected but not yet started:



(Yngve 1960)

Yngve's depth

Number of constituents expected but not yet started:

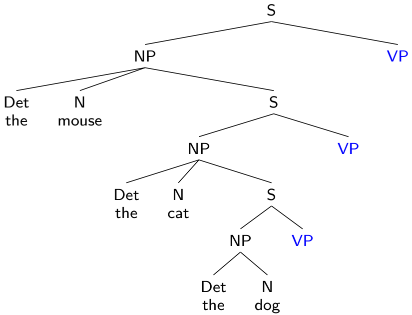


- Unlike (center-)self-embedding, right-embedding doesn't create such large lists of expected constituents (because the expected stuff is all part of **one** constituent).
- But left-embedding does.

(Yngve 1960)

Yngve's depth

Number of constituents expected but not yet started:



- Unlike (center-)self-embedding, right-embedding doesn't create such large lists of expected constituents (because the expected stuff is all part of **one** constituent).
- But left-embedding does.

- Yngve's theory was set within — perhaps justified by — a procedural story, but we can arguably detach it from that and treat depth as just another property of trees.

(Yngve 1960)

Interpretation functions for "complexity"

What are some other interpretation functions?

- number of nodes
- ratio of total nodes to terminal nodes (Miller and Chomsky 1963)
- degree of self-embedding (Miller and Chomsky 1963)
- "depth" of memory required by a top-down parser (Yngve 1960)
- minimal attachment, late closure, etc.? (Frazier and Clifton 1996)

Reaching conclusions about grammars

complexity metric $+$ grammar $\longrightarrow$ prediction

Typically, arguments hold the grammar fixed and present evidence in favour of a metric.

Reaching conclusions about grammars

complexity metric $+$ grammar $\longrightarrow$ prediction

Typically, arguments hold the grammar fixed and present evidence in favour of a metric.

We can flip this around: hold the metric fixed and present evidence in favour of a grammar.

> *If we accept — as I do — . . . that the rules of grammar enter into the processing mechanisms, then evidence concerning production, recognition, recall, and language use in general can be expected (in principle) to have bearing on the investigation of rules of grammar, on what is sometimes called "grammatical competence" or "knowledge of language".*
>
> *(Chomsky 1980: pp.200-201)*

Reaching conclusions about grammars

$$\text{complexity metric} \quad + \quad \text{grammar} \quad \longrightarrow \quad \text{prediction}$$

Example: hold self-embedding fixed as the complexity metric.

Reaching conclusions about grammars

complexity metric $+$ grammar $\longrightarrow$ prediction

Example: hold self-embedding fixed as the complexity metric.

(4) That [the food that [John ordered] tasted good] pleased him.

(5) That [that [the food was good] pleased John] surprised Mary.

Grammar question: Does a relative clause have a node labeled S?

Reaching conclusions about grammars

complexity metric $+$ grammar $\longrightarrow$ prediction

Example: hold self-embedding fixed as the complexity metric.

(4) That [the food that [John ordered] tasted good] pleased him.

(5) That [that [the food was good] pleased John] surprised Mary.

Grammar question: Does a relative clause have a node labeled S?

Proposed answer	(4) structure	(5) structure	Prediction
Yes	$\ldots[_S \ldots[_S \ldots]\ ]$	$\ldots[_S \ldots[_S \ldots]\ ]$	(4) & (5) same
No	$\ldots[_S \ldots[_{RC} \ldots]\ ]$	$\ldots[_S \ldots[_S \ldots]\ ]$	(5) harder

Reaching conclusions about grammars

complexity metric $+$ grammar $\longrightarrow$ prediction

Example: hold self-embedding fixed as the complexity metric.

(4) That [the food that [John ordered] tasted good] pleased him.

(5) That [that [the food was good] pleased John] surprised Mary.

Grammar question: Does a relative clause have a node labeled S?

Proposed answer	(4) structure	(5) structure	Prediction
Yes	$\ldots[_S \ldots[_S \ldots]\ ]$	$\ldots[_S \ldots[_S \ldots]\ ]$	(4) & (5) same
No	$\ldots[_S \ldots[_{RC} \ldots]\ ]$	$\ldots[_S \ldots[_S \ldots]\ ]$	(5) harder

Conclusion: The fact that (5) is harder supports the “No” answer.

Outline

Derivations and representations

Question

But these metrics are all properties of a final, fully-constructed tree.
How can anything like this be sensitive to differences in the derivational operations that build these trees? (e.g. TAG vs. MG, whether move is re-merge)

Interpretation functions for "complexity"

What are some other interpretation functions?

- number of nodes
- ratio of total nodes to terminal nodes (Miller and Chomsky 1963)
- degree of self-embedding (Miller and Chomsky 1963)
- "depth" of memory required by a top-down parser (Yngve 1960)
- minimal attachment, late closure, etc.? (Frazier and Clifton 1996)
- "nature, number and complexity of" transformations (Miller and Chomsky 1963)

"nature, number and complexity of the grammatical transformations involved"

The psychological plausibility of a transformational model of the language user would be strengthened, of course, if it could be shown that our performance on tasks requiring an appreciation of the structure of transformed sentences is ***some function of the nature, number and complexity of the grammatical transformations involved****.*

(Miller and Chomsky 1963: p.481)

Derivations and representations

Question

But these metrics are all properties of a final, fully-constructed tree.
How can anything like this be sensitive to differences in the derivational operations that build these trees? (e.g. TAG vs. MG, whether move is re-merge)

Derivations and representations

Question

But these metrics are all properties of a final, fully-constructed tree.
How can anything like this be sensitive to differences in the derivational operations that build these trees? (e.g. TAG vs. MG, whether move is re-merge)

Answer

The relevant objects on which the interpretation functions are defined encode a complete derivational history.

Derivations and representations

Question

But these metrics are all properties of a final, fully-constructed tree.
How can anything like this be sensitive to differences in the derivational operations that build these trees? (e.g. TAG vs. MG, whether move is re-merge)

Answer

The relevant objects on which the interpretation functions are defined encode a complete derivational history.

e.g. The function which, given a complete "recipe" for carrying out a derivation, returns the number of movement steps called for by the recipe.

Full derivation recipes?

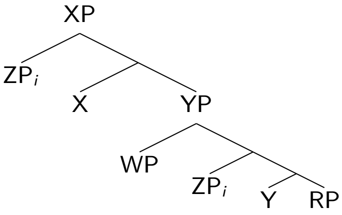
Are the inputs to these functions **really** full derivation recipes?

For minimalist syntax it's hard to tell, because the final derived object very often uniquely identifies a derivational history/recipe.

Full derivation recipes?

Are the inputs to these functions **really** full derivation recipes?

For minimalist syntax it's hard to tell, because the final derived object very often uniquely identifies a derivational history/recipe.



- merge Y with RP
- merge the result with ZP
- merge the result with WP
- merge X with the result
- move ZP

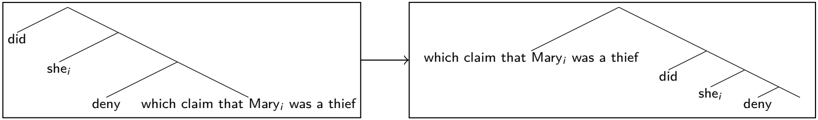
Full derivation recipes?

A few cases reveal that (we must all be already assuming that) it's full derivations/recipes that count.

Full derivation recipes?

A few cases reveal that (we must all be already assuming that) it's full derivations/recipes that count.

(6) * Which claim [that Mary$_i$ was a thief] did she$_i$ deny?

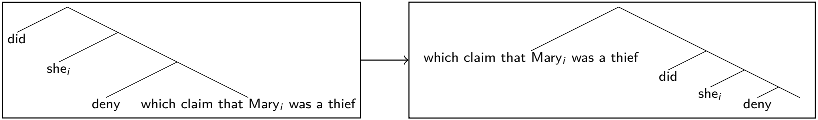


(7) Which claim [that Mary$_i$ made] did she$_i$ deny?

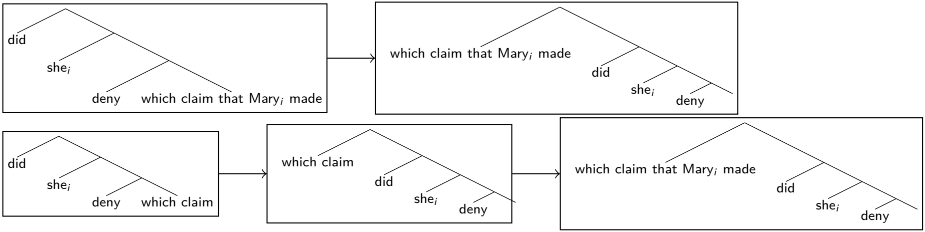
Full derivation recipes?

A few cases reveal that (we must all be already assuming that) it's full derivations/recipes that count.

(6) * Which claim [that Mary$_i$ was a thief] did she$_i$ deny?



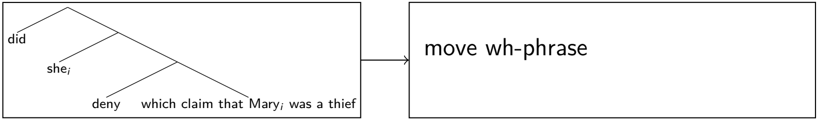
(7) Which claim [that Mary$_i$ made] did she$_i$ deny?



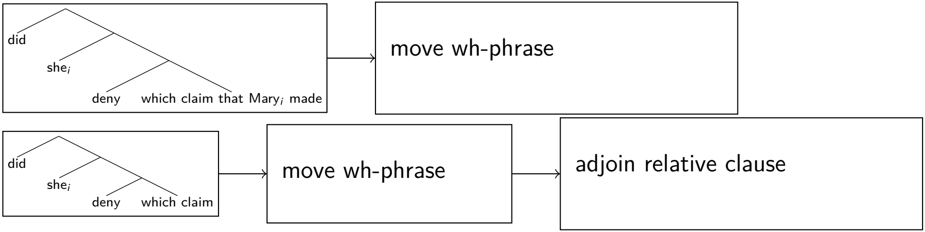
Full derivation recipes?

A few cases reveal that (we must all be already assuming that) it's full derivations/recipes that count.

(6) * Which claim [that Mary$_i$ was a thief] did she$_i$ deny?



(7) Which claim [that Mary$_i$ made] did she$_i$ deny?



Full derivation recipes?

Also:

- subjacency effects without traces
- compare categorial grammar

Full derivation recipes?

Also:

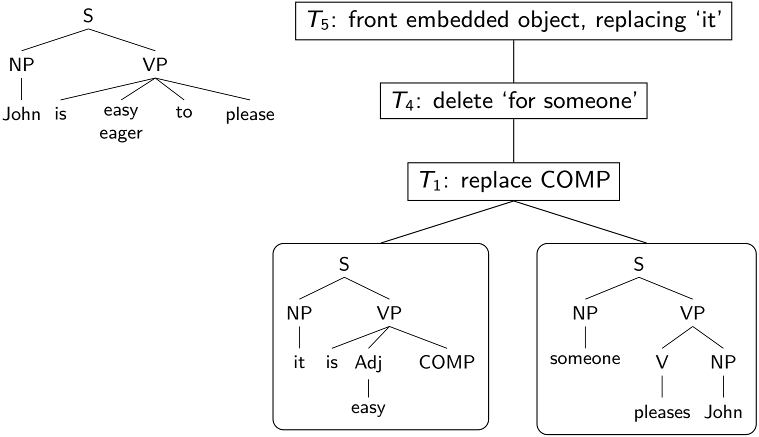
- subjacency effects without traces
- compare categorial grammar

And this is not a new idea!

> *[The perceptual model] will utilize the full resources of the transformational grammar to provide a structural description, consisting of a set of P-markers and a transformational history*
>
> *Miller and Chomsky (1963: p.480)*

Full derivation recipes?

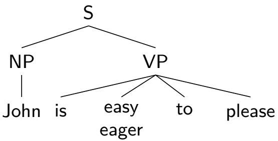
> *[The perceptual model] will utilize the full resources of the transformational grammar to provide a structural description, consisting of a set of P-markers and a transformational history*
>
> *Miller and Chomsky (1963: p.480)*

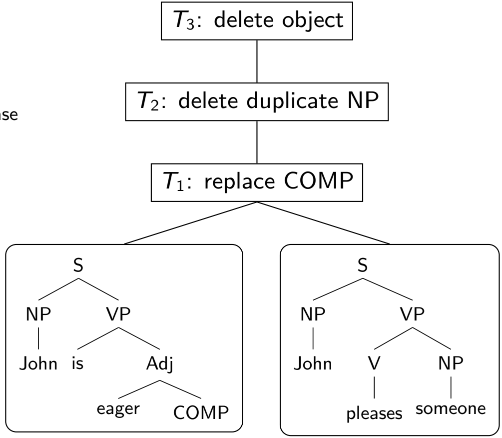


Full derivation recipes?

[The perceptual model] will utilize the full resources of the transformational grammar to provide a structural description, consisting of a set of P-markers and a transformational history

Miller and Chomsky (1963: p.480)



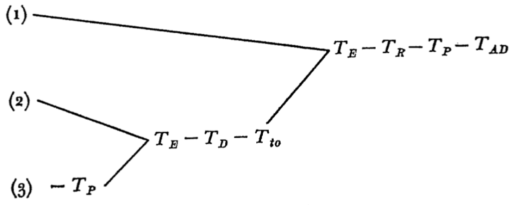


Full derivation recipes?

(4) the man who persuaded John to be examined by a specialist was fired

The "transformational history" of (4) by which it is derived from its basis might be represented, informally, by the diagram (5).

(5)



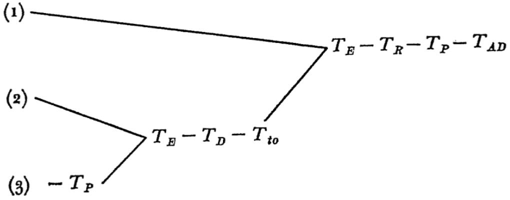
(Chomsky 1965)

Full derivation recipes?

(4) the man who persuaded John to be examined by a specialist was fired

The "transformational history" of (4) by which it is derived from its basis might be represented, informally, by the diagram (5).

(5)



Differences these days:

- We'll have things like merge and move at the internal nodes instead of T_P, T_E, etc.
- We'll have lexical items at the leaves rather than base-derived trees.

(Chomsky 1965)

Outline

Surprisal and entropy reduction

Why these complexity metrics?

- Partly just for concreteness, to give us a goal.
- They are formalism neutral to a degree that others aren't.
- They are mechanism neutral (Marr level one).
- The pieces of the puzzle that we need to get there (e.g. probabilities) seem likely to be usable in other ways.

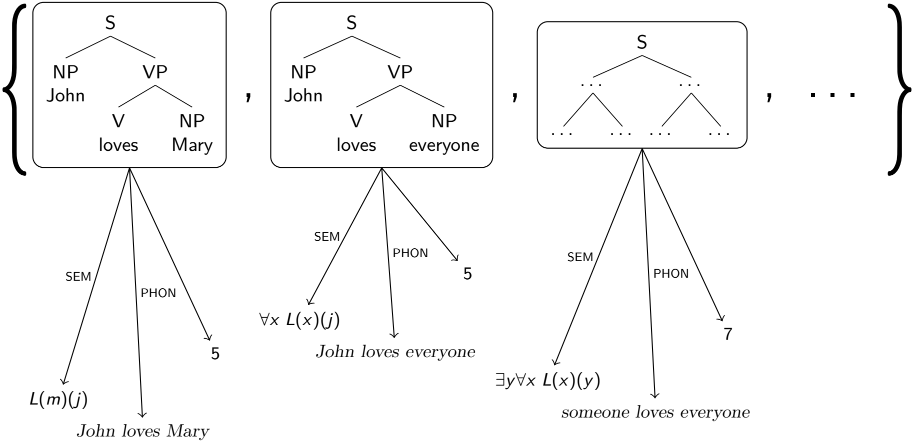
Surprisal and entropy reduction

Why these complexity metrics?

- Partly just for concreteness, to give us a goal.
- They are formalism neutral to a degree that others aren't.
- They are mechanism neutral (Marr level one).
- The pieces of the puzzle that we need to get there (e.g. probabilities) seem likely to be usable in other ways.

John Hale, Cornell Univ.

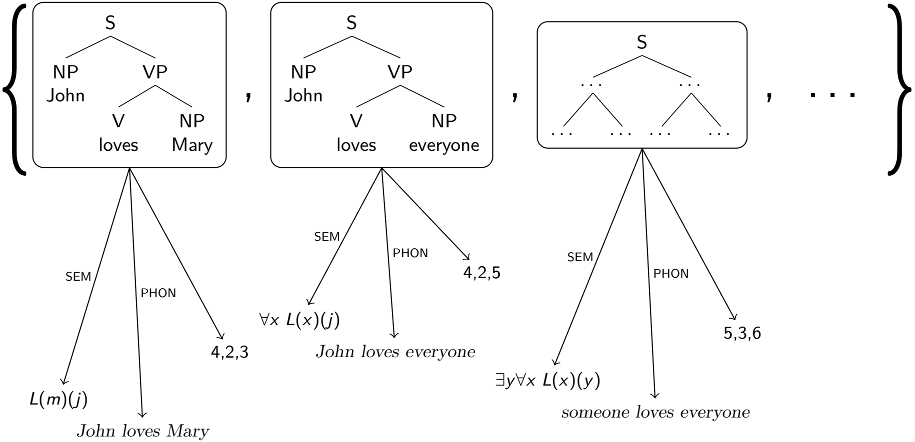
Interpretation functions



Caveats:

- Maybe we're interested in the finite specification of the set
- Maybe there's no clear line between observable and not
- Maybe some evidence is based on relativities among interpretations

Interpretation functions



Caveats:

- Maybe we're interested in the finite specification of the set
- Maybe there's no clear line between observable and not
- Maybe some evidence is based on relativities among interpretations

Surprisal

Given a sentence $w_1 w_2 \ldots w_n$:

$$\text{surprisal at } w_i = -\log P(W_i = w_i \mid W_1 = w_1, W_2 = w_2, \ldots, W_{i-1} = w_{i-1})$$

Surprisal

0.4	John ran
0.15	John saw it
0.05	John saw them
0.25	Mary ran
0.1	Mary saw it
0.05	Mary saw them

What predictions can we make about the difficulty of comprehending 'John saw it'?

Surprisal

0.4	John ran
0.15	John saw it
0.05	John saw them
0.25	Mary ran
0.1	Mary saw it
0.05	Mary saw them

What predictions can we make about the difficulty of comprehending 'John saw it'?

$$\begin{aligned}\text{surprisal at 'John'} &= -\log P(W_1 = \text{John}) \\ &= -\log(0.4 + 0.15 + 0.05) \\ &= -\log 0.6 \\ &= 0.74\end{aligned}$$

Surprisal

0.4	John ran
0.15	John saw it
0.05	John saw them
0.25	Mary ran
0.1	Mary saw it
0.05	Mary saw them

What predictions can we make about the difficulty of comprehending 'John saw it'?

$$\begin{aligned}\text{surprisal at 'John'} &= -\log P(W_1 = \text{John}) \\ &= -\log(0.4 + 0.15 + 0.05) \\ &= -\log 0.6 \\ &= 0.74\end{aligned}$$

$$\begin{aligned}\text{surprisal at 'saw'} &= -\log P(W_2 = \text{saw} \mid W_1 = \text{John}) \\ &= -\log \frac{0.15 + 0.05}{0.4 + 0.15 + 0.05} \\ &= -\log 0.33 \\ &= 1.58\end{aligned}$$

Surprisal

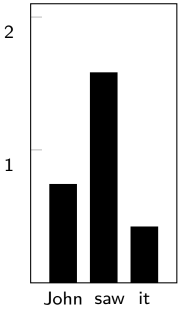
0.4	John ran
0.15	John saw it
0.05	John saw them
0.25	Mary ran
0.1	Mary saw it
0.05	Mary saw them

What predictions can we make about the difficulty of comprehending 'John saw it'?

$$\begin{aligned}\text{surprisal at 'John'} &= -\log P(W_1 = \text{John}) \\ &= -\log(0.4 + 0.15 + 0.05) \\ &= -\log 0.6 \\ &= 0.74\end{aligned}$$

$$\begin{aligned}\text{surprisal at 'saw'} &= -\log P(W_2 = \text{saw} \mid W_1 = \text{John}) \\ &= -\log \frac{0.15 + 0.05}{0.4 + 0.15 + 0.05} \\ &= -\log 0.33 \\ &= 1.58\end{aligned}$$

$$\begin{aligned}\text{surprisal at 'it'} &= -\log P(W_3 = \text{it} \mid W_1 = \text{John}, W_2 = \text{saw}) \\ &= -\log \frac{0.15}{0.15 + 0.05} \\ &= -\log 0.75 \\ &= 0.42\end{aligned}$$



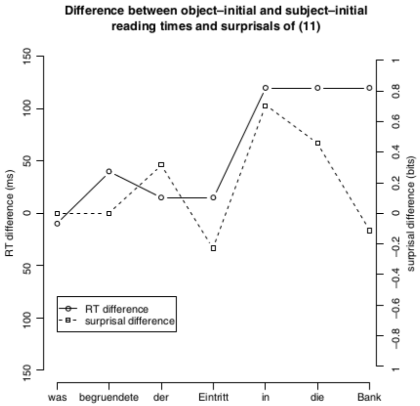
Accurate predictions made by surprisal

(Hale 2001)

Accurate predictions made by surprisal

(8) The reporter [who ____ attacked the senator] left the room. (easier)

(9) The reporter [who the senator attacked ____] left the room. (harder)



(Levy 2008)

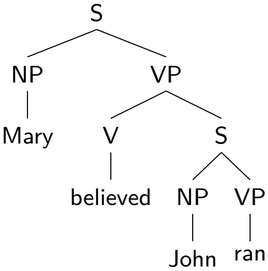
An important distinction

Using surprisal as a complexity metric says nothing about the form of the knowledge that the language comprehender is using!

- We're asking "what's the probability of w_i, given that we've seen $w_1 \ldots w_{i-1}$ in the past".
- This does not mean that the comprehender's knowledge takes the form of answers to this kind of question.
- The linear nature of the metric reflects the task, not the knowledge being probed.

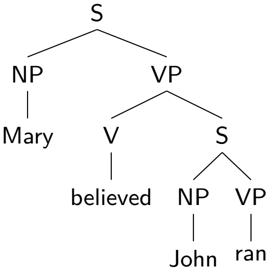
Probabilistic CFGs

1.0	S $\rightarrow$ NP VP
0.3	NP $\rightarrow$ John
0.7	NP $\rightarrow$ Mary
0.2	VP $\rightarrow$ ran
0.5	VP $\rightarrow$ V NP
0.3	VP $\rightarrow$ V S
0.4	V $\rightarrow$ believed
0.6	V $\rightarrow$ knew



Probabilistic CFGs

1.0	S → NP VP
0.3	NP → John
0.7	NP → Mary
0.2	VP → ran
0.5	VP → V NP
0.3	VP → V S
0.4	V → believed
0.6	V → knew



$$P(\text{Mary believed John ran}) = 1.0 \times 0.7 \times 0.3 \times 0.4 \times 1.0 \times 0.3 \times 0.2$$
$$= 0.00504$$

Surprisal with probabilistic CFGs

Goal: Calculate step-by-step surprisal values for 'Mary believed John ran'

surprisal at 'John' $= -\log P(W_3 = \text{John} \mid W_1 = \text{Mary}, W_2 = \text{believed})$

Surprisal with probabilistic CFGs

Goal: Calculate step-by-step surprisal values for ‘Mary believed John ran’

surprisal at ‘John’ $= -\log P(W_3 = \text{John} \mid W_1 = \text{Mary}, W_2 = \text{believed})$

0.098	**Mary believed** Mary
0.042	**Mary believed** John
0.012348	**Mary believed** Mary knew Mary
0.01176	**Mary believed** Mary ran
0.008232	**Mary believed** Mary believed Mary
0.005292	**Mary believed** Mary knew John
0.005292	**Mary believed** John knew Mary
0.00504	**Mary believed** John ran
...	...

Surprisal with probabilistic CFGs

Goal: Calculate step-by-step surprisal values for 'Mary believed John ran'

surprisal at 'John' $= -\log P(W_3 = \text{John} \mid W_1 = \text{Mary}, W_2 = \text{believed})$

0.098	**Mary believed** Mary
0.042	**Mary believed** John
0.012348	**Mary believed** Mary knew Mary
0.01176	**Mary believed** Mary ran
0.008232	**Mary believed** Mary believed Mary
0.005292	**Mary believed** Mary knew John
0.005292	**Mary believed** John knew Mary
0.00504	**Mary believed** John ran
...	...

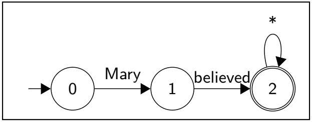
There are an infinite number of derivations consistent with input at each point!

$$\text{surprisal at 'John'} = -\log P(W_3 = \text{John} \mid W_1 = \text{Mary}, W_2 = \text{believed})$$
$$= -\log \frac{0.042 + 0.005292 + 0.00504 + \dots}{0.098 + 0.042 + 0.12348 + 0.01176 + 0.008232 + \dots}$$

Intersection grammars

1.0	S $\rightarrow$ NP VP
0.3	NP $\rightarrow$ John
0.7	NP $\rightarrow$ Mary
0.2	VP $\rightarrow$ ran
0.5	VP $\rightarrow$ V NP
0.3	VP $\rightarrow$ V S
0.4	V $\rightarrow$ believed
0.6	V $\rightarrow$ knew

$\cap$

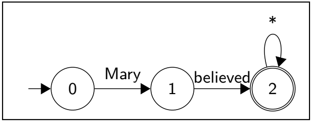


$= \quad G_2$

Intersection grammars

1.0	S → NP VP
0.3	NP → John
0.7	NP → Mary
0.2	VP → ran
0.5	VP → V NP
0.3	VP → V S
0.4	V → believed
0.6	V → knew

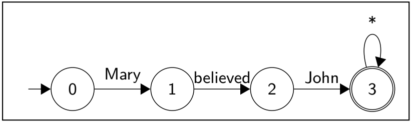
$\cap$



$= G_2$

1.0	S → NP VP
0.3	NP → John
0.7	NP → Mary
0.2	VP → ran
0.5	VP → V NP
0.3	VP → V S
0.4	V → believed
0.6	V → knew

$\cap$

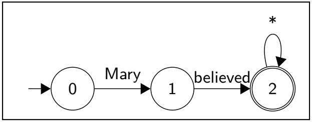


$= G_3$

Intersection grammars

1.0	S → NP VP
0.3	NP → John
0.7	NP → Mary
0.2	VP → ran
0.5	VP → V NP
0.3	VP → V S
0.4	V → believed
0.6	V → knew

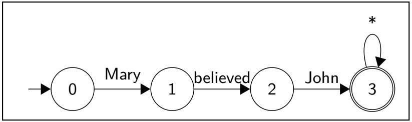
$\cap$



$= \quad G_2$

1.0	S → NP VP
0.3	NP → John
0.7	NP → Mary
0.2	VP → ran
0.5	VP → V NP
0.3	VP → V S
0.4	V → believed
0.6	V → knew

$\cap$

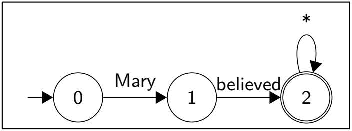


$= \quad G_3$

$$\begin{aligned}
\text{surprisal at `John'} &= -\log P(W_3 = \text{John} \mid W_1 = \text{Mary}, W_2 = \text{believed}) \\
&= -\log \frac{\text{total weight in } G_3}{\text{total weight in } G_2} \\
&= -\log \frac{0.0672}{0.224} \\
&= 1.74
\end{aligned}$$

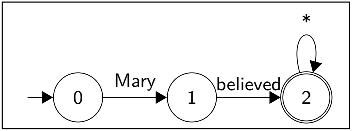
Grammar intersection example (simple)

1.0	S → NP VP
0.3	NP → John
0.7	NP → Mary
0.2	VP → ran
0.5	VP → V NP
0.3	VP → V S
0.4	V → believed
0.6	V → knew



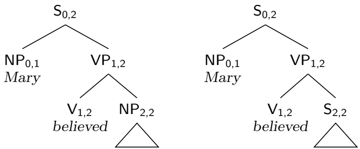
Grammar intersection example (simple)

1.0	S → NP VP
0.3	NP → John
0.7	NP → Mary
0.2	VP → ran
0.5	VP → V NP
0.3	VP → V S
0.4	V → believed
0.6	V → knew



1.0	$S_{0,2}$	→	$NP_{0,1}$ $VP_{1,2}$
0.7	$NP_{0,1}$	→	Mary
0.5	$VP_{1,2}$	→	$V_{1,2}$ $NP_{2,2}$
0.3	$VP_{1,2}$	→	$V_{1,2}$ $S_{2,2}$
0.4	$V_{1,2}$	→	believed

1.0	$S_{2,2}$	→	$NP_{2,2}$ $VP_{2,2}$
0.3	$NP_{2,2}$	→	John
0.7	$NP_{2,2}$	→	Mary
0.2	$VP_{2,2}$	→	ran
0.5	$VP_{2,2}$	→	$V_{2,2}$ $NP_{2,2}$
0.3	$VP_{2,2}$	→	$V_{2,2}$ $S_{2,2}$
0.4	$V_{2,2}$	→	believed
0.6	$V_{2,2}$	→	knew

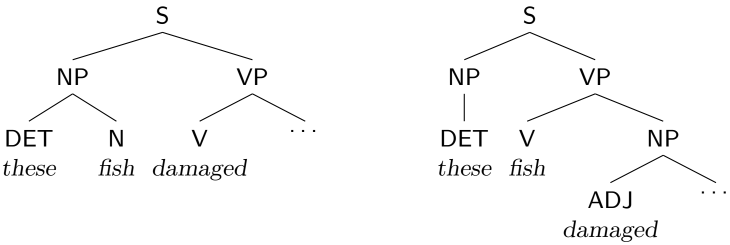


NB: Total weight in this grammar is not one! (What is it? Start symbol is $S_{0,2}$.)
Each derivation has the weight "it" had in the original grammar.

Grammar intersection example (more complicated)

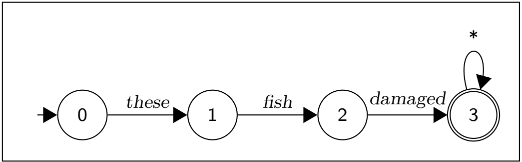
S	→	NP VP	V	→	*fish*
VP	→	V NP	V	→	*damaged*
NP	→	DET	DET	→	*these*
NP	→	DET N	N	→	*fish*
NP	→	ADJ N	ADJ	→	*damaged*

These fish damaged ...

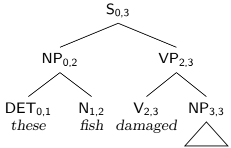


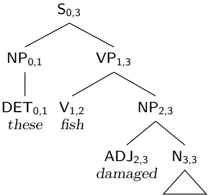
Grammar intersection example (more complicated)

S	→	NP VP	V	→	*fish*
VP	→	V NP	V	→	*damaged*
NP	→	DET	DET	→	*these*
NP	→	DET N	N	→	*fish*
NP	→	ADJ N	ADJ	→	*damaged*



$S_{0,3}$	→	$NP_{0,2}$ $VP_{2,3}$	$S_{0,3}$	→	$NP_{0,1}$ $VP_{1,3}$	$NP_{3,3}$	→	$ADJ_{3,3}$ $N_{3,3}$
$NP_{0,2}$	→	$DET_{0,1}$ $N_{1,2}$	$NP_{0,1}$	→	$DET_{0,1}$	$NP_{3,3}$	→	$DET_{3,3}$ $N_{3,3}$
$VP_{2,3}$	→	$V_{2,3}$ $NP_{3,3}$	$VP_{1,3}$	→	$V_{1,2}$ $NP_{2,3}$	$NP_{3,3}$	→	$DET_{3,3}$
$DET_{0,1}$	→	*these*	$NP_{2,3}$	→	$ADJ_{2,3}$ $N_{3,3}$	$N_{3,3}$	→	*fish*
$N_{1,2}$	→	*fish*	$V_{1,2}$	→	*fish*	$DET_{3,3}$	→	*these*
$V_{2,3}$	→	*damaged*	$ADJ_{2,3}$	→	*damaged*	$ADJ_{3,3}$	→	*damaged*

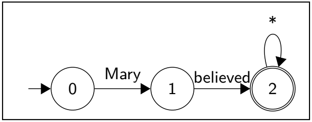




Intersection grammars

1.0	S → NP VP
0.3	NP → John
0.7	NP → Mary
0.2	VP → ran
0.5	VP → V NP
0.3	VP → V S
0.4	V → believed
0.6	V → knew

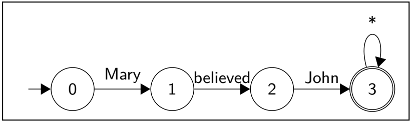
$\cap$



$= G_2$

1.0	S → NP VP
0.3	NP → John
0.7	NP → Mary
0.2	VP → ran
0.5	VP → V NP
0.3	VP → V S
0.4	V → believed
0.6	V → knew

$\cap$



$= G_3$

$$\begin{aligned}\text{surprisal at `John'} &= -\log P(W_3 = \text{John} \mid W_1 = \text{Mary}, W_2 = \text{believed}) \\ &= -\log \frac{\text{total weight in } G_3}{\text{total weight in } G_2} \\ &= -\log \frac{0.0672}{0.224} \\ &= 1.74\end{aligned}$$

Computing sum of weights in a grammar (“partition function”)

$$Z(A) = \sum_{A \to \alpha} \Big(p(A \to \alpha) \cdot Z(\alpha) \Big)$$
$$Z(\epsilon) = 1$$
$$Z(a\beta) = Z(\beta)$$
$$Z(B\beta) = Z(B) \cdot Z(\beta) \qquad \text{where } \beta \neq \epsilon$$

(Nederhof and Satta 2008)

1.0	S → NP VP
0.3	NP → John
0.7	NP → Mary
0.2	VP → ran
0.5	VP → V NP
0.4	V → believed
0.6	V → knew

$$Z(\text{V}) = 0.4 + 0.6 = 1.0$$
$$Z(\text{NP}) = 0.3 + 0.7 = 1.0$$
$$Z(\text{VP}) = 0.2 + (0.5 \cdot Z(\text{V}) \cdot Z(\text{NP}))$$
$$= 0.2 + (0.5 \cdot 1.0 \cdot 1.0) = 0.7$$
$$Z(\text{S}) = 1.0 \cdot Z(\text{NP}) \cdot Z(\text{VP})$$
$$= 0.7$$

Computing sum of weights in a grammar ("partition function")

$$Z(A) = \sum_{A \to \alpha} \Big(p(A \to \alpha) \cdot Z(\alpha) \Big)$$
$$Z(\epsilon) = 1$$
$$Z(a\beta) = Z(\beta)$$
$$Z(B\beta) = Z(B) \cdot Z(\beta) \qquad \text{where } \beta \neq \epsilon$$

(Nederhof and Satta 2008)

1.0	S → NP VP
0.3	NP → John
0.7	NP → Mary
0.2	VP → ran
0.5	VP → V NP
0.4	V → believed
0.6	V → knew

$$Z(\text{V}) = 0.4 + 0.6 = 1.0$$
$$Z(\text{NP}) = 0.3 + 0.7 = 1.0$$
$$Z(\text{VP}) = 0.2 + (0.5 \cdot Z(\text{V}) \cdot Z(\text{NP}))$$
$$= 0.2 + (0.5 \cdot 1.0 \cdot 1.0) = 0.7$$
$$Z(\text{S}) = 1.0 \cdot Z(\text{NP}) \cdot Z(\text{VP})$$
$$= 0.7$$

1.0	S → NP VP
0.3	NP → John
0.7	NP → Mary
0.2	VP → ran
0.5	VP → V NP
0.3	VP → V S
0.4	V → believed
0.6	V → knew

$$Z(\text{V}) = 0.4 + 0.6 = 1.0$$
$$Z(\text{NP}) = 0.3 + 0.7 = 1.0$$
$$Z(\text{VP}) = 0.2 + (0.5 \cdot Z(\text{V}) \cdot Z(\text{NP})) + (0.3 \cdot Z(\text{V}) \cdot Z(\text{S}))$$
$$Z(\text{S}) = 1.0 \cdot Z(\text{NP}) \cdot Z(\text{VP})$$

Things to know

Technical facts about CFGs:

- Can intersect with a "prefix FSA"
- Can compute the total weight (and the entropy)

(Hale 2006)

Things to know

Technical facts about CFGs:

- Can intersect with a "prefix FSA"
- Can compute the total weight (and the entropy)

More generally:

- Intersecting a grammar with a prefix produces a new grammar which is a representation of the comprehender's sentence-medial state
- So we can construct a sequence of grammars which represents the comprehender's sequence of knowledge-states
- Ask "what changes" (or "how much changes", etc.) at each step

The general approach is compatible with many very different grammar formalisms (any grammar formalism?) — provided the technical tricks can be pulled off.

(Hale 2006)

Looking ahead

Wouldn't it be nice if we could do all that for minimalist syntax?

The average syntax paper shows illustrative derivations, not a fragment.

What would we need?

- An explicit characterization of the set of possible derivations
- A way to "intersect" that with a prefix
- A way to define probability distributions over the possibilities

This will require certain idealizations. (But what's new?)

Part 2

Minimalist Grammars

Outline

Outline

Wait a minute!

"I thought the whole point was deciding between candidate sets of primitive derivational operations! Isn't it begging the question to set everything in stone at the beginning like this?"

Wait a minute!

"I thought the whole point was deciding between candidate sets of primitive derivational operations! Isn't it begging the question to set everything in stone at the beginning like this?"

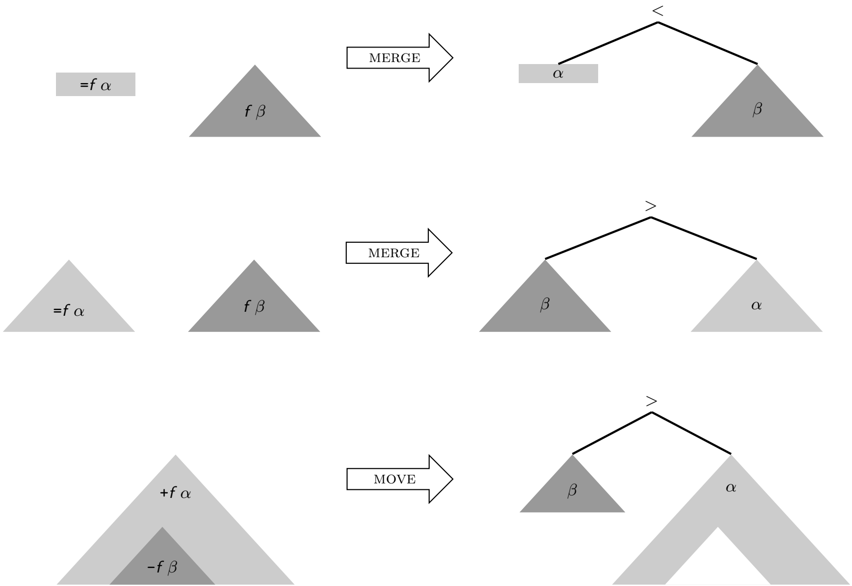
- We're not setting this in stone — we will look at alternatives.
- But we need a concrete starting point so that we can make the differences concrete.
- What's coming up is meant as a relatively neutral/"mainstream" starting point.

Minimalist Grammars

Defining a grammar in the MG formalism is defining a set *Lex* of lexical items

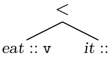
- A lexical item is a string with a sequence of features.
  e.g. *like* :: `=d =d v`, *mary* :: `d`, *who* :: `d -wh`
- Generates the closure of the $Lex \subset Expr$ under two derivational operations:
  - MERGE : $Expr \times Expr \xrightarrow{\text{partial}} Expr$
  - MOVE : $Expr \xrightarrow{\text{partial}} Expr$
- Each feature encodes a requirement that must be met by applying a particular derivational operation.
  - MERGE checks `=f` and `f`
  - MOVE checks `+f` and `-f`
- A derived expression is complete when it has only a single feature remaining unchecked.

Merge and move

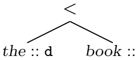


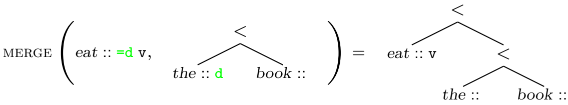
Examples

MERGE (*eat* :: =d v, *it* :: d) =

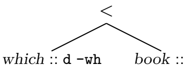


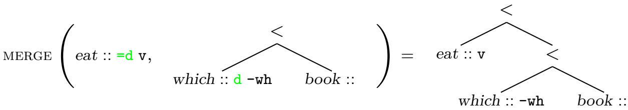
MERGE (*the* :: =n d, *book* :: n) =



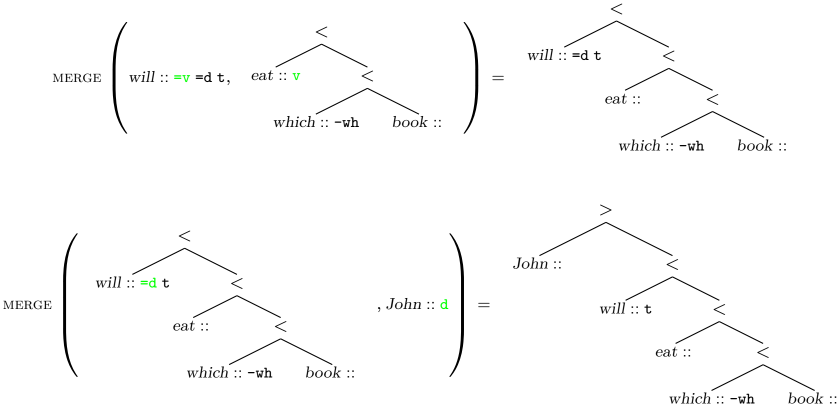


MERGE (*which* :: =n d -wh, *book* :: n) =

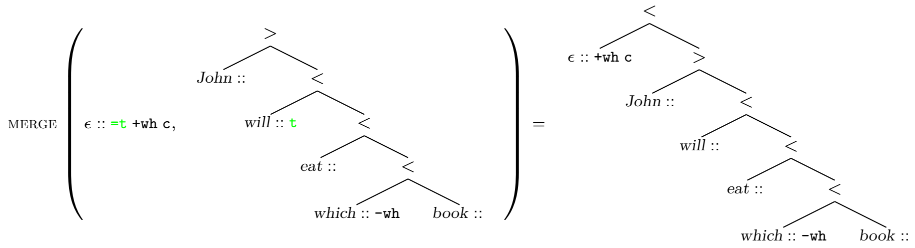


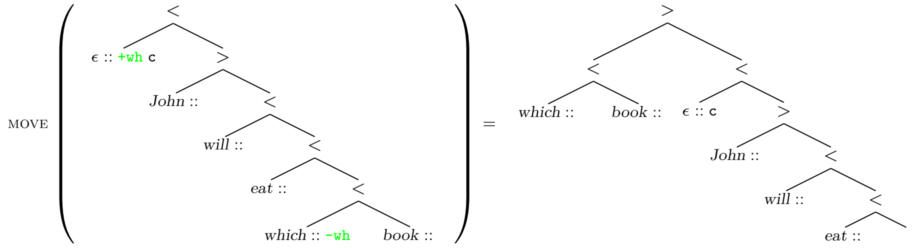


Examples

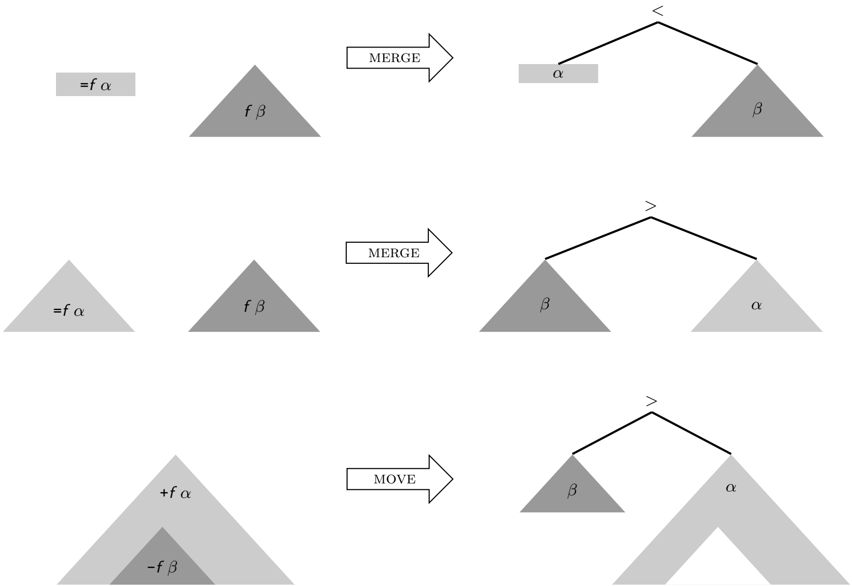

MERGE
will :: =v =d t,
eat :: v
which :: -wh
book ::
=
will :: =d t
eat ::
which :: -wh
book ::
MERGE
will :: =d t
eat ::
which :: -wh
book ::
, John :: d
=
John ::
will :: t
eat ::
which :: -wh
book ::

Examples


MERGE
ϵ :: =t +wh c,
>
John ::
<
will :: t
<
eat ::
<
which :: -wh
book ::
=
<
ϵ :: +wh c
>
John ::
<
will ::
<
eat ::
<
which :: -wh
book ::


MOVE
<
ϵ :: +wh c
>
John ::
<
will ::
<
eat ::
<
which :: -wh
book ::
=
>
<
which ::
book ::
<
ϵ :: c
>
John ::
<
will ::
<
eat ::

Merge and move



Definitions

$$\text{MERGE}\big(e_1[=f\ \alpha], e_2[f\ \beta]\big) = \begin{cases} [_< \ e_1[\alpha]\ e_2[\beta]] & \text{if } e_1[=f\ \alpha] \in \mathit{Lex} \\ [_> \ e_2[\beta]\ e_1[\alpha]] & \text{otherwise} \end{cases}$$

$$\text{MOVE}\big(e_1[+f\ \alpha]\big) = [_> \ e_2[\beta]\ e_1'[\alpha]]$$

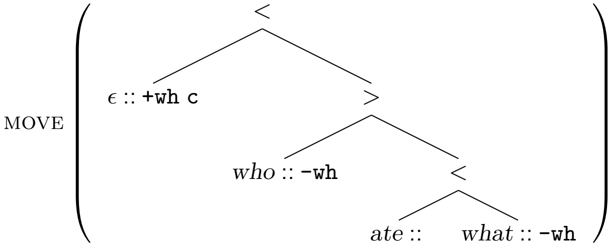
where $e_2[-f\ \beta]$ is a unique subtree of $e_1[+f\ \alpha]$

and e_1' is like e_1 but with $e_2[-f\ \beta]$ replaced by an empty leaf node

Shortest Move Constraint

How do we know which subtree should be displaced when we apply MOVE?

By stipulation, there can only ever be one candidate. This is the Shortest Move Constraint (SMC).

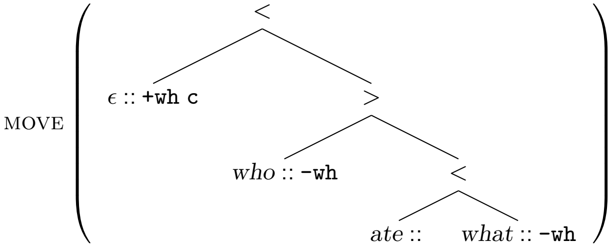


is **undefined**

Shortest Move Constraint

How do we know which subtree should be displaced when we apply MOVE?

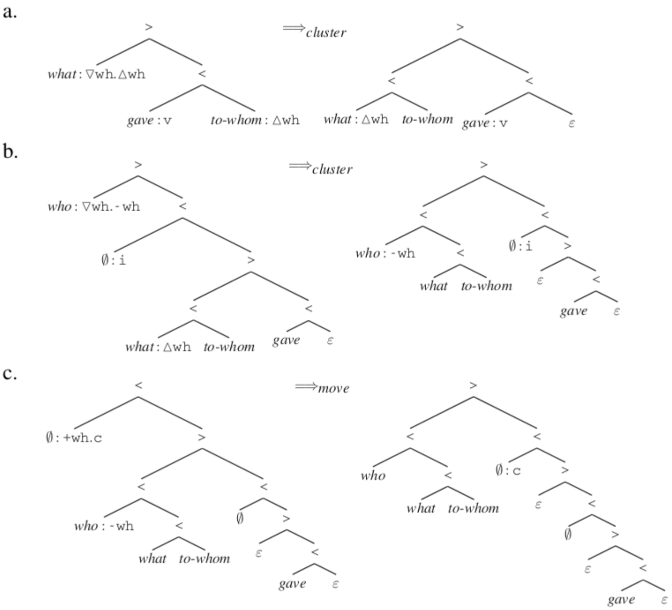
By stipulation, there can only ever be one candidate. This is the Shortest Move Constraint (SMC).



is **undefined**

Q: Multiple wh-movement?
A: Clustering!

(7)


a.
what : ∇wh.∆wh
gave : v
to-whom : ∆wh
⟹cluster
what : ∆wh
to-whom
gave : v
ε
b.
who : ∇wh.-wh
∅ : i
what : ∆wh
to-whom
gave
ε
⟹cluster
who : -wh
what
to-whom
∅ : i
ε
gave
ε
c.
∅ : +wh.c
who : -wh
what
to-whom
∅
ε
gave
ε
⟹move
who
what
to-whom
∅ : c
ε
∅
ε
gave
ε

Notation

`=d v` or `=dp vp`?

Notation

=d v or =dp vp?

Categorial grammar:

- Primitive symbols for "complete" things, e.g. S, NP
- Derived symbols for "incomplete" things, e.g. S\NP
- Lexical category can specify "what's missing"

Notation

`=d v` or `=dp vp`?

Categorial grammar:

- Primitive symbols for "complete" things, e.g. S, NP
- Derived symbols for "incomplete" things, e.g. S\NP
- Lexical category can specify "what's missing"

Traditional X-bar theory:

- Primitive symbols for "incomplete" things, e.g. V, T
- Derived symbols for "complete" things, e.g. VP, TP ($= V''$, T'')
- Separate subcategorization info specifies "what's missing"

Notation

`=d v` or `=dp vp`?

Categorial grammar:

- Primitive symbols for "complete" things, e.g. S, NP
- Derived symbols for "incomplete" things, e.g. S\NP
- Lexical category can specify "what's missing"

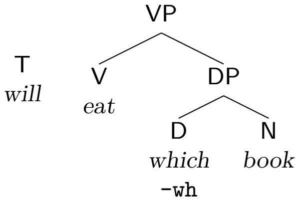
Traditional X-bar theory:

- Primitive symbols for "incomplete" things, e.g. V, T
- Derived symbols for "complete" things, e.g. VP, TP ($=$ V$''$, T$''$)
- Separate subcategorization info specifies "what's missing"

MGs:

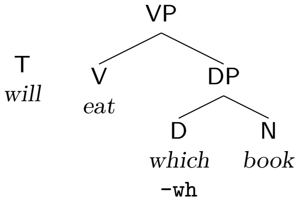
- Primitive symbols for "complete" things, like CG
- So `t` means "a complete projection of T", not "a T head"

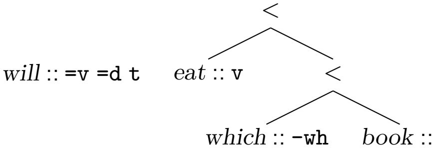
Notation comparison



	Conventional notation
'eat which book' is a VP	VP label on root
'which book' must move	`-wh` on 'which'
'will' combines with a VP	implicit

Notation comparison





	Conventional notation	MG notation
'eat which book' is a VP	VP label on root	v on 'eat'
'which book' must move	-wh on 'which'	-wh on 'which'
'will' combines with a VP	implicit	=v on 'will'

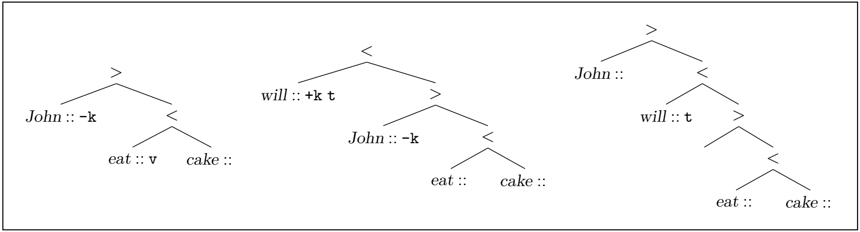
Outline

A Minimalist Grammar

cake :: `d`	*what* :: `d -wh`
John :: `d -k`	*who* :: `d -k -wh`
eat :: `=d =d v`	ϵ :: `=t +wh c`
will :: `=v +k t`	ϵ :: `=t c`

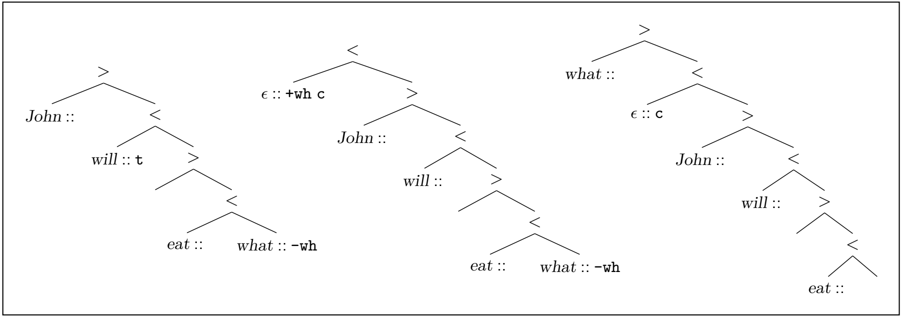
A Minimalist Grammar

cake :: d	*what* :: d -wh
John :: d -k	*who* :: d -k -wh
eat :: =d =d v	ϵ :: =t +wh c
will :: =v +k t	ϵ :: =t c



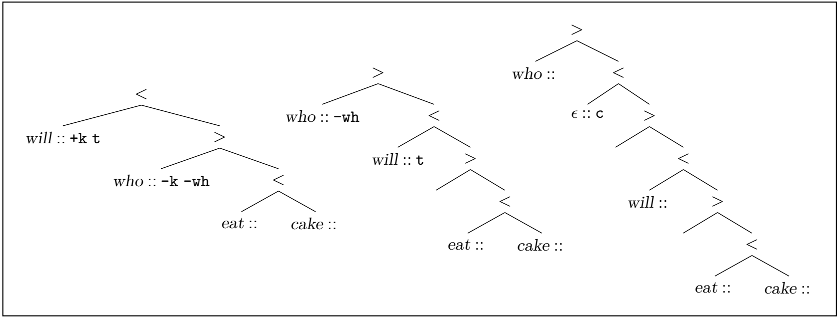
A Minimalist Grammar

cake :: `d`	*what* :: `d -wh`
John :: `d -k`	*who* :: `d -k -wh`
eat :: `=d =d v`	ϵ :: `=t +wh c`
will :: `=v +k t`	ϵ :: `=t c`



A Minimalist Grammar

cake :: `d`	*what* :: `d -wh`
John :: `d -k`	*who* :: `d -k -wh`
eat :: `=d =d v`	ϵ :: `=t +wh c`
will :: `=v +k t`	ϵ :: `=t c`

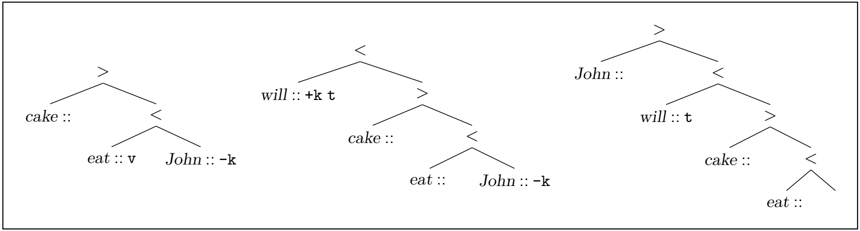


A Minimalist Grammar . . . which overgenerates

cake :: `d`	*what* :: `d -wh`
John :: `d -k`	*who* :: `d -k -wh`
eat :: `=d =d v`	ϵ :: `=t +wh c`
will :: `=v +k t`	ϵ :: `=t c`

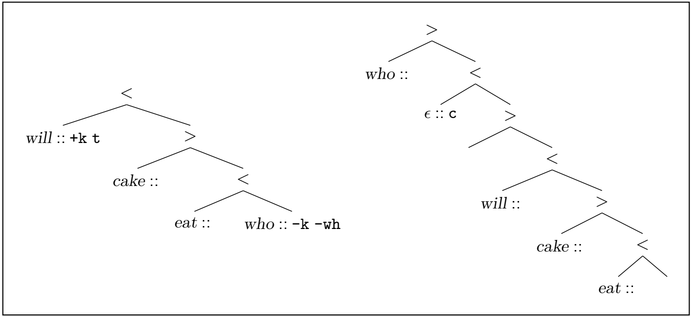
A Minimalist Grammar . . . which overgenerates

cake :: d	*what* :: d -wh
John :: d -k	*who* :: d -k -wh
eat :: =d =d v	ϵ :: =t +wh c
will :: =v +k t	ϵ :: =t c



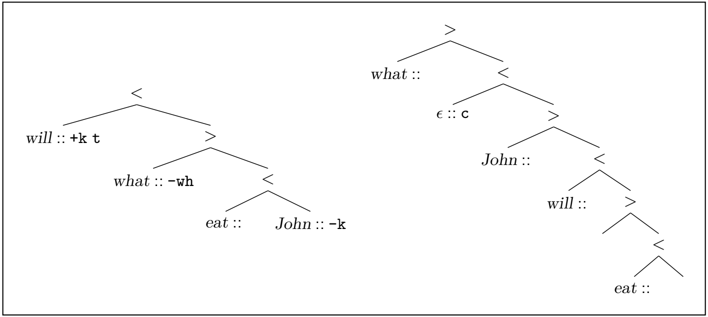
A Minimalist Grammar . . . which overgenerates

cake :: `d`	*what* :: `d -wh`
John :: `d -k`	*who* :: `d -k -wh`
eat :: `=d =d v`	ϵ :: `=t +wh c`
will :: `=v +k t`	ϵ :: `=t c`



A Minimalist Grammar . . . which overgenerates

cake :: `d`	*what* :: `d -wh`
John :: `d -k`	*who* :: `d -k -wh`
eat :: `=d =d v`	ϵ :: `=t +wh c`
will :: `=v +k t`	ϵ :: `=t c`



A Minimalist Grammar . . . which overgenerates

cake :: `d`	*what* :: `d -wh`
John :: `d -k`	*who* :: `d -k -wh`
eat :: `=d =d v`	ϵ :: `=t +wh c`
will :: `=v +k t`	ϵ :: `=t c`

John will eat cake	*John will cake eat*
what John will eat	*what John will eat*
who will eat cake	*who will cake eat*

A Minimalist Grammar . . . which overgenerates

cake :: `d`	*what* :: `d -wh`
John :: `d -k`	*who* :: `d -k -wh`
eat :: `=d =d v`	ϵ :: `=t +wh c`
will :: `=v +k t`	ϵ :: `=t c`

John will eat cake	*John will cake eat*
what John will eat	*what John will eat*
who will eat cake	*who will cake eat*

S	$\rightarrow$	NP VP	VP	$\rightarrow$	V NP
NP	$\rightarrow$	*John*	VP	$\rightarrow$	*runs*
NP	$\rightarrow$	*Mary*	VP	$\rightarrow$	*walks*
			V	$\rightarrow$	*loves*

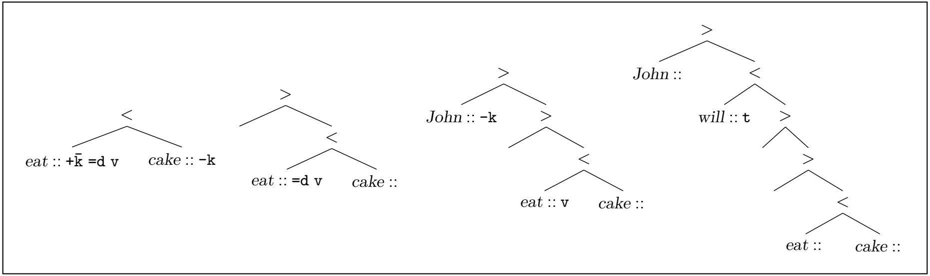
John runs	*Mary runs*
John walks	*Mary walks*
John loves John	*Mary loves John*
John loves Mary	*Mary loves Mary*

First solution: covert movement/agree

cake :: d -k	*what* :: d -k -wh
John :: d -k	*who* :: d -k -wh
eat :: =d $+\bar{k}$ =d v	ϵ :: =t +wh c
will :: =v +k t	ϵ :: =t c

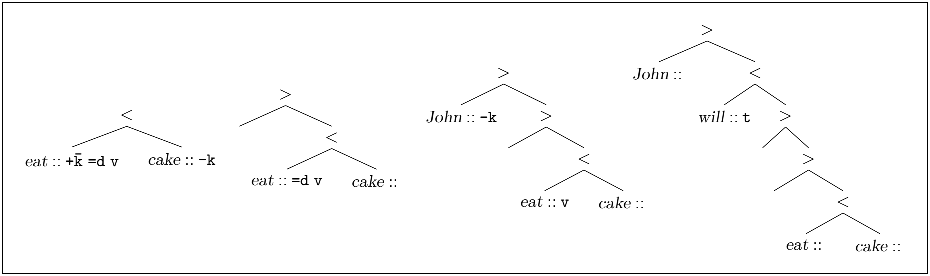
First solution: covert movement/agree

cake :: d -k	*what* :: d -k -wh
John :: d -k	*who* :: d -k -wh
eat :: =d +$\bar{\text{k}}$ =d v	ϵ :: =t +wh c
will :: =v +k t	ϵ :: =t c



First solution: covert movement/agree

cake :: d -k	*what* :: d -k -wh
John :: d -k	*who* :: d -k -wh
eat :: =d +$\bar{k}$ =d v	ϵ :: =t +wh c
will :: =v +k t	ϵ :: =t c



Note order of features on *eat*!

Second solution

Separate `d` into `subj` and `obj`

cake :: `obj`	*what* :: `obj -wh`
John :: `subj -k`	*who* :: `subj -k -wh`
eat :: `=obj =subj v`	ϵ :: `=t +wh c`
will :: `=v +k t`	ϵ :: `=t c`

Problem "solved":

John will eat cake
what John will eat
who will eat cake

Outline

Adding embedded clauses

cake :: `obj`	*what* :: `obj -wh`	*think* :: `=c =subj v`
John :: `subj -k`	*who* :: `subj -k -wh`	*ask* :: `=q =subj v`
eat :: `=obj =subj v`	ϵ :: `=t +wh q`	*Mary* :: `subj -k`
will :: `=v +k t`	ϵ :: `=t c`	

Adding embedded clauses

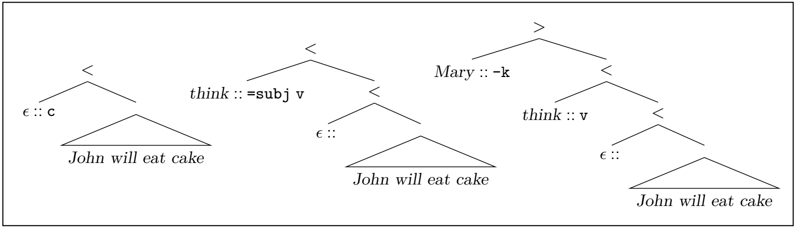
cake :: `obj`	*what* :: `obj -wh`	*think* :: `=c =subj v`
John :: `subj -k`	*who* :: `subj -k -wh`	*ask* :: `=q =subj v`
eat :: `=obj =subj v`	ϵ :: `=t +wh q`	*Mary* :: `subj -k`
will :: `=v +k t`	ϵ :: `=t c`	

John will eat cake	*Mary will think John will eat cake*	...
what John will eat	*what Mary will think John will eat*	...
who will eat cake	*who Mary will think will eat cake*	...

Adding embedded clauses

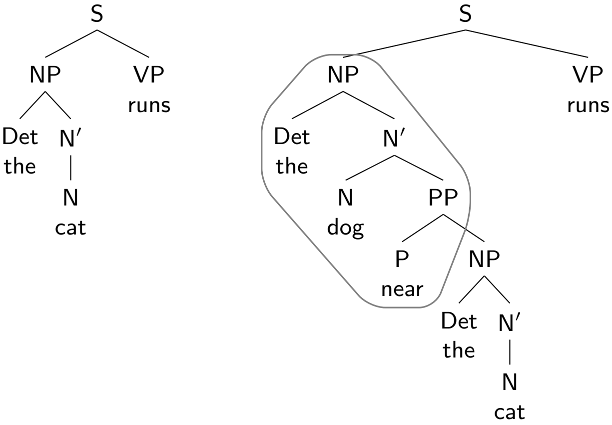
cake :: `obj`	*what* :: `obj -wh`	*think* :: `=c =subj v`
John :: `subj -k`	*who* :: `subj -k -wh`	*ask* :: `=q =subj v`
eat :: `=obj =subj v`	ϵ :: `=t +wh q`	*Mary* :: `subj -k`
will :: `=v +k t`	ϵ :: `=t c`	

John will eat cake	*Mary will think John will eat cake*	...
what John will eat	*what Mary will think John will eat*	...
who will eat cake	*who Mary will think will eat cake*	...



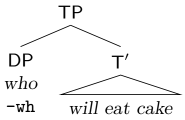
Reminder: "Loops" in a CFG

S	$\rightarrow$	NP VP	VP	$\rightarrow$	*runs*
NP	$\rightarrow$	Det N′	Det	$\rightarrow$	*the*
N′	$\rightarrow$	N	N	$\rightarrow$	*dog*
N′	$\rightarrow$	N PP	N	$\rightarrow$	*cat*
PP	$\rightarrow$	P NP	P	$\rightarrow$	*near*

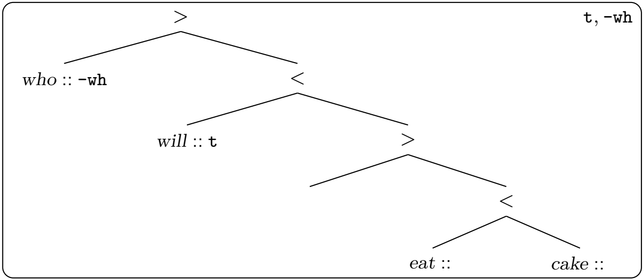


Which extensions create “loops”?

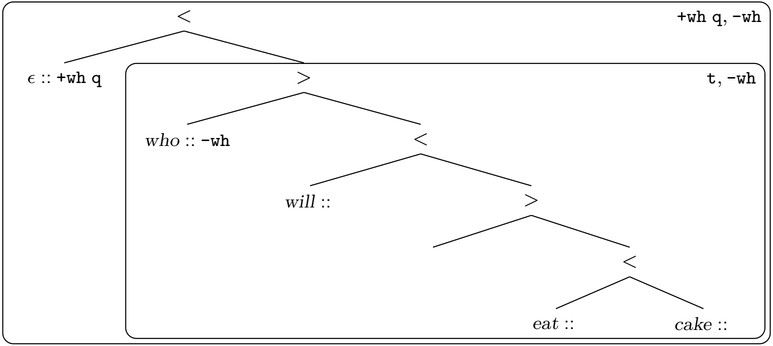
Starting point:



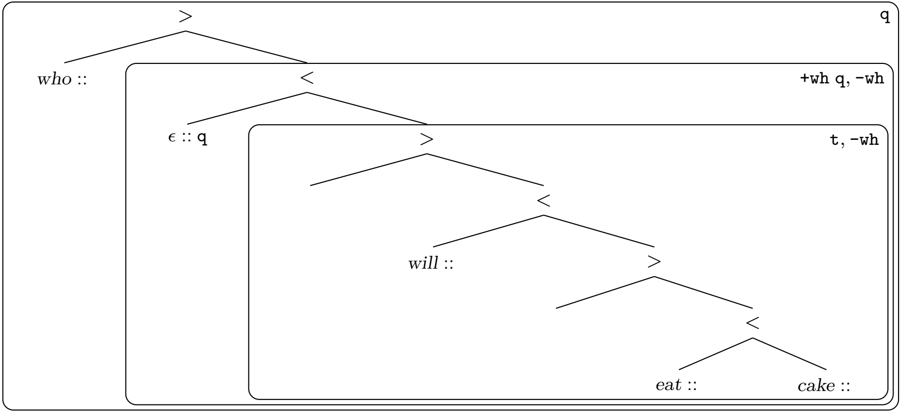
A simple, non-looping completion



A simple, non-looping completion

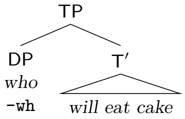


A simple, non-looping completion



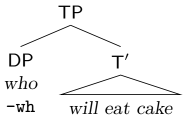
Which extensions create "loops"?

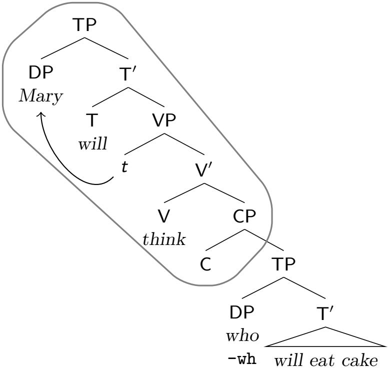
Starting point:



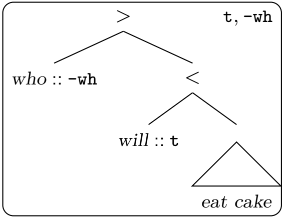
Which extensions create "loops"?

Starting point:

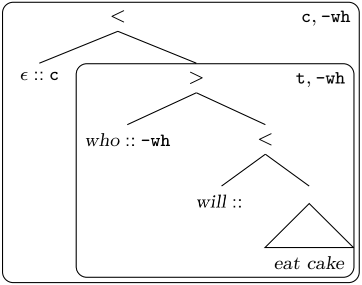




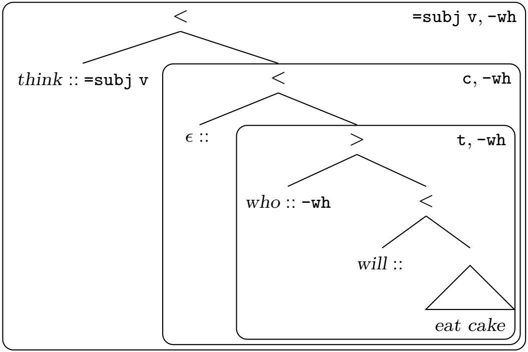
Extending with *Mary will think . . .*



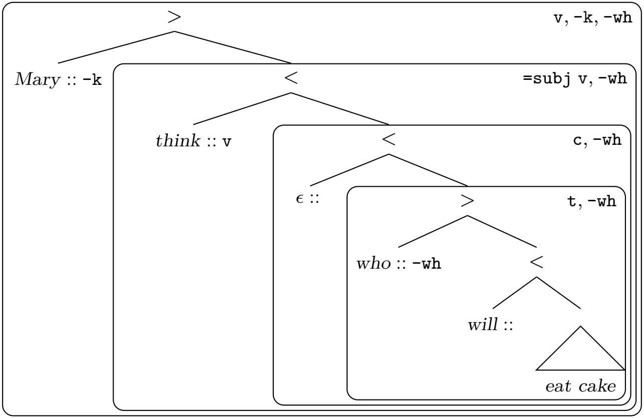
Extending with *Mary will think ...*



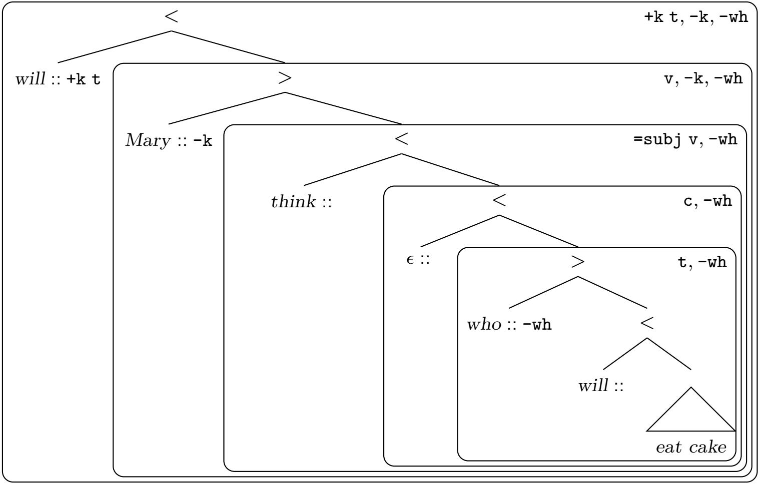
Extending with *Mary will think ...*



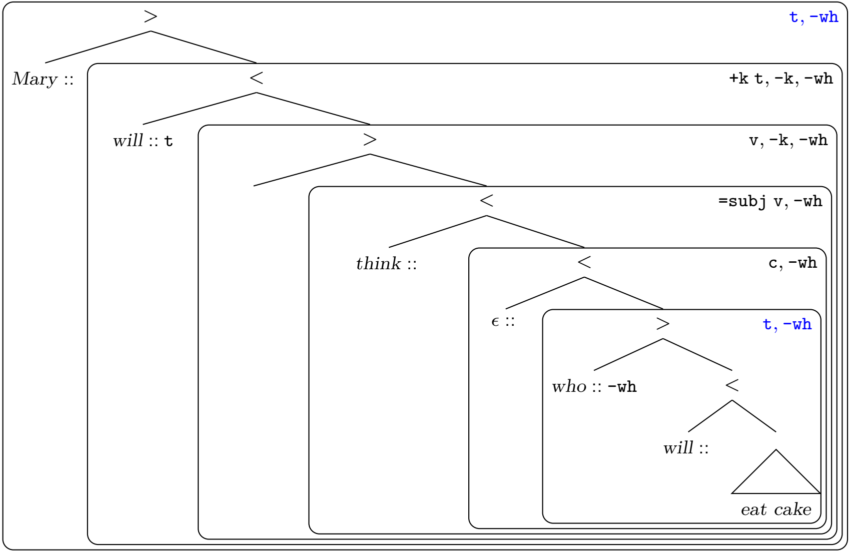
Extending with *Mary will think* . . .



Extending with *Mary will think ...*

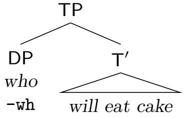


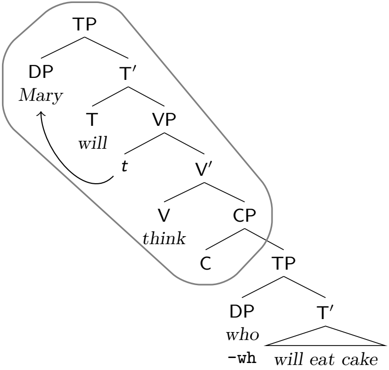
Extending with *Mary will think . . .*



Which extensions create “loops”?

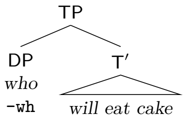
Starting point:

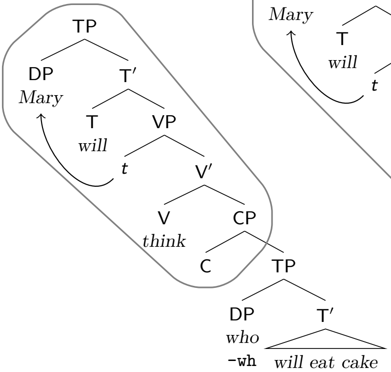


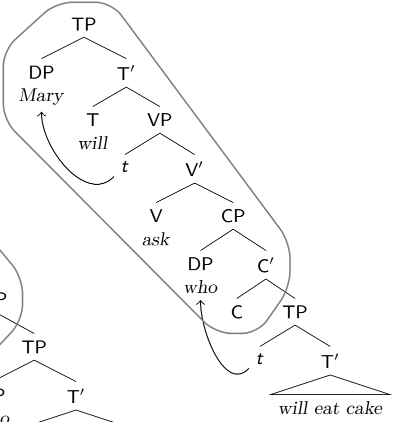


Which extensions create "loops"?

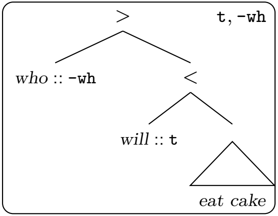
Starting point:



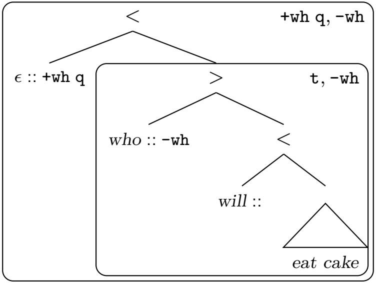




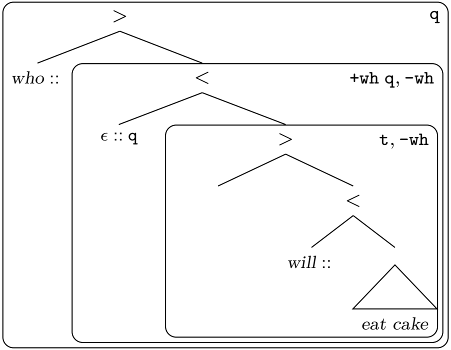
Extending with *Mary will ask . . .*



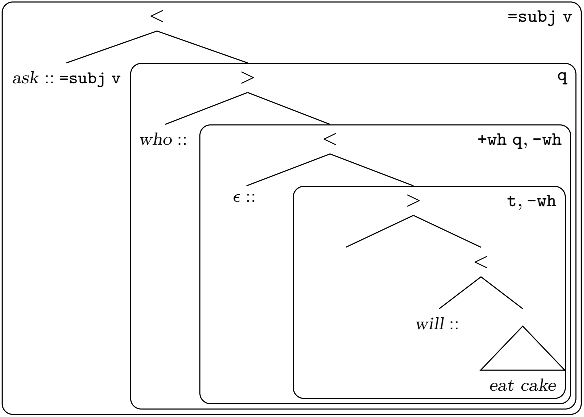
Extending with *Mary will ask ...*



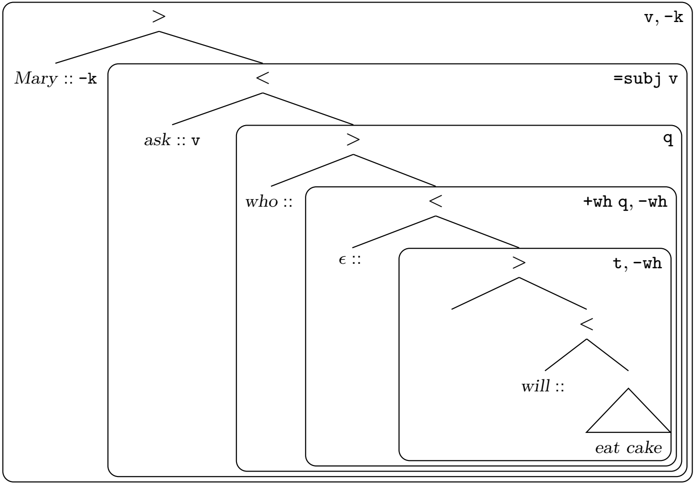
Extending with *Mary will ask ...*



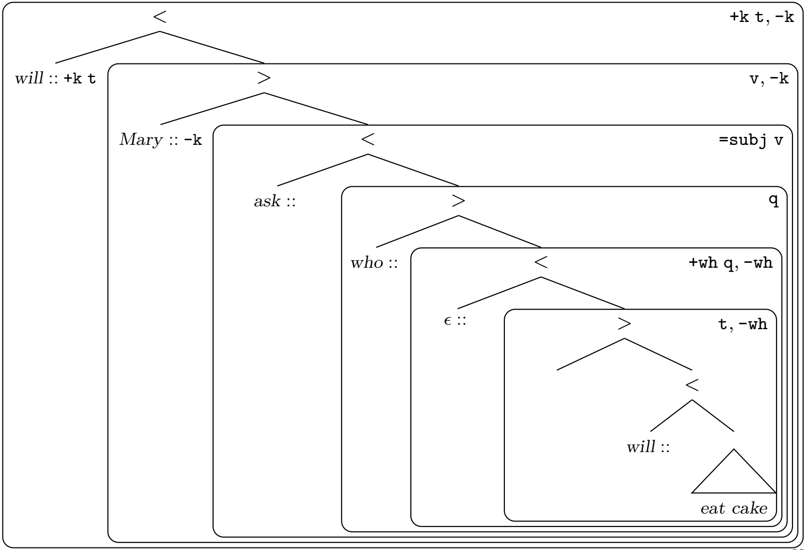
Extending with *Mary will ask ...*



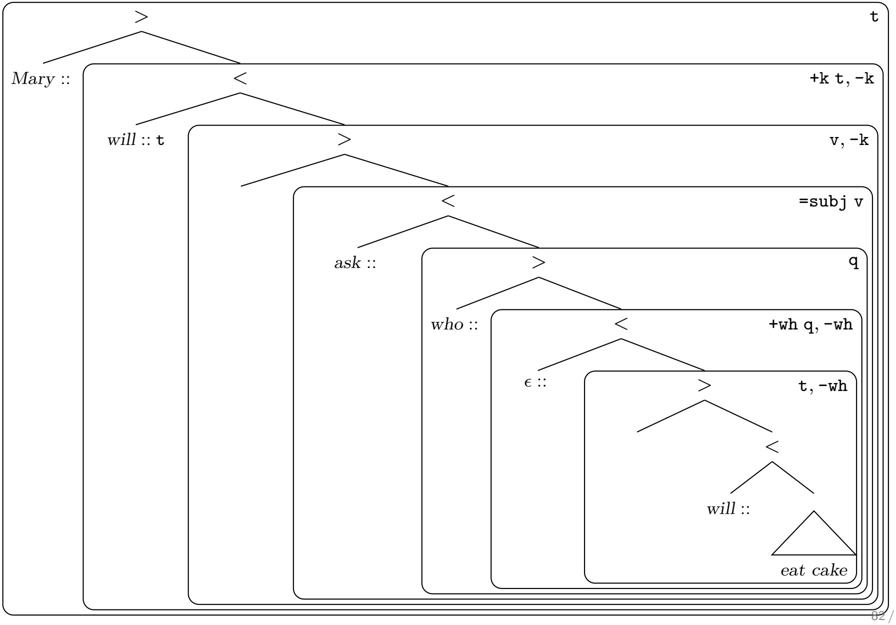
Extending with *Mary will ask* . . .



Extending with *Mary will ask . . .*

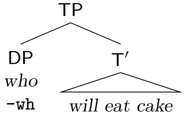


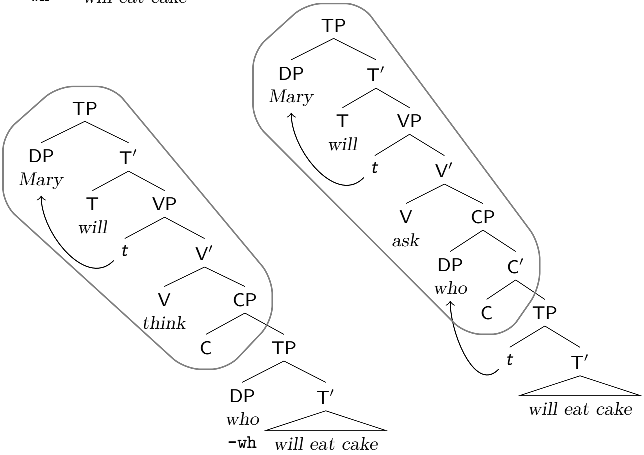
Extending with *Mary will ask . . .*



Which extensions create “loops”?

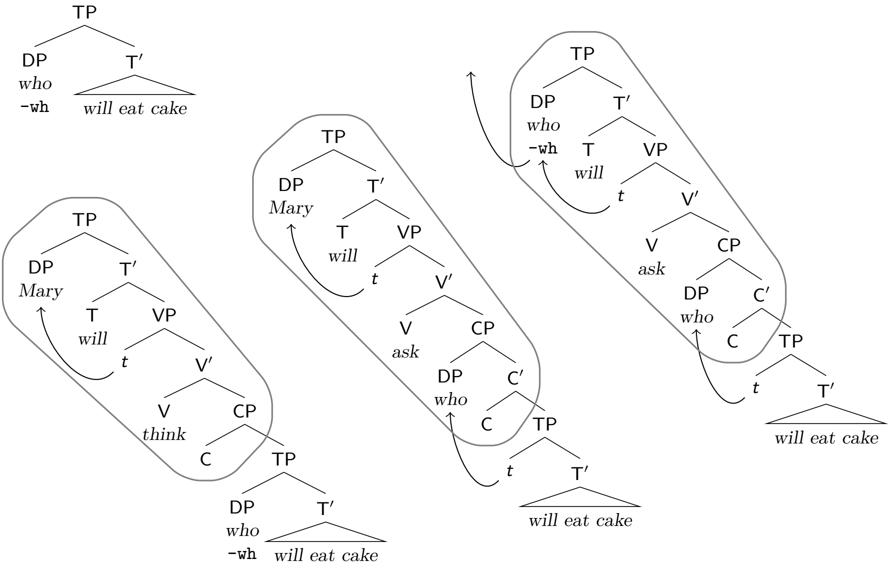
Starting point:



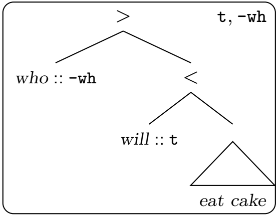


Which extensions create "loops"?

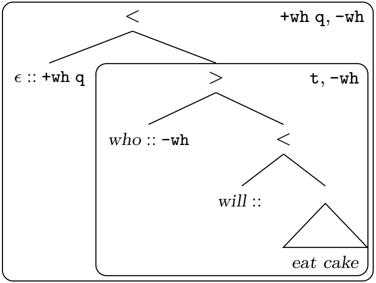
Starting point:



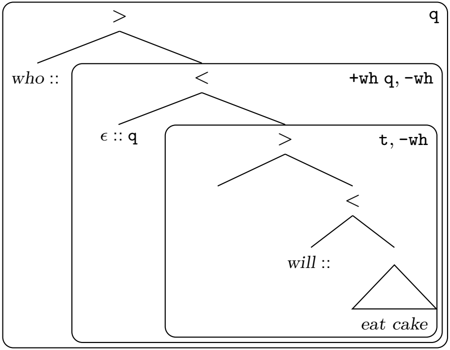
Extending with *who will ask . . .*



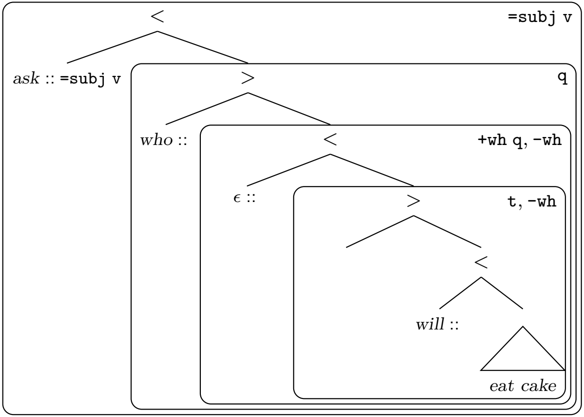
Extending with *who will ask ...*



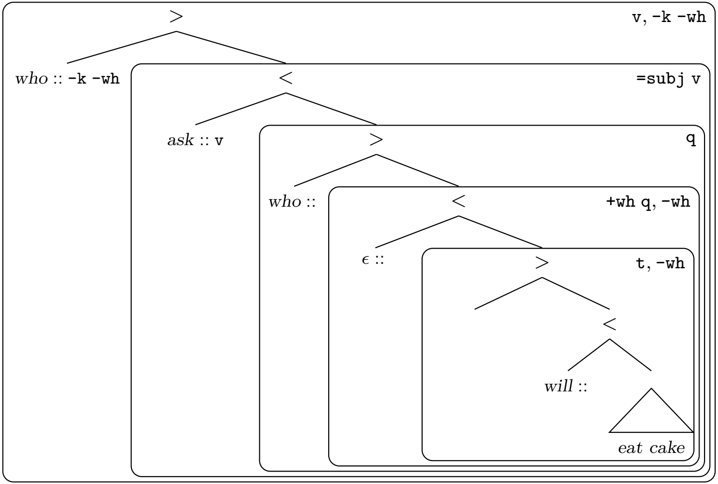
Extending with *who will ask ...*



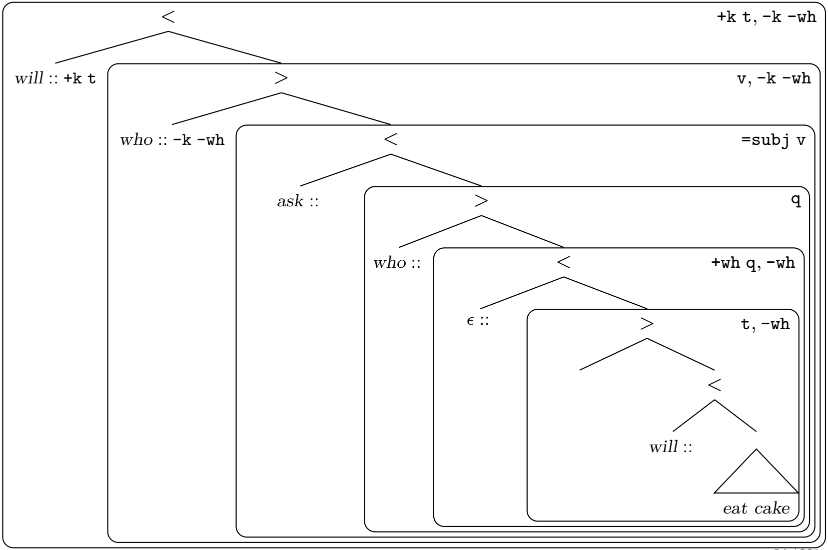
Extending with *who will ask* . . .



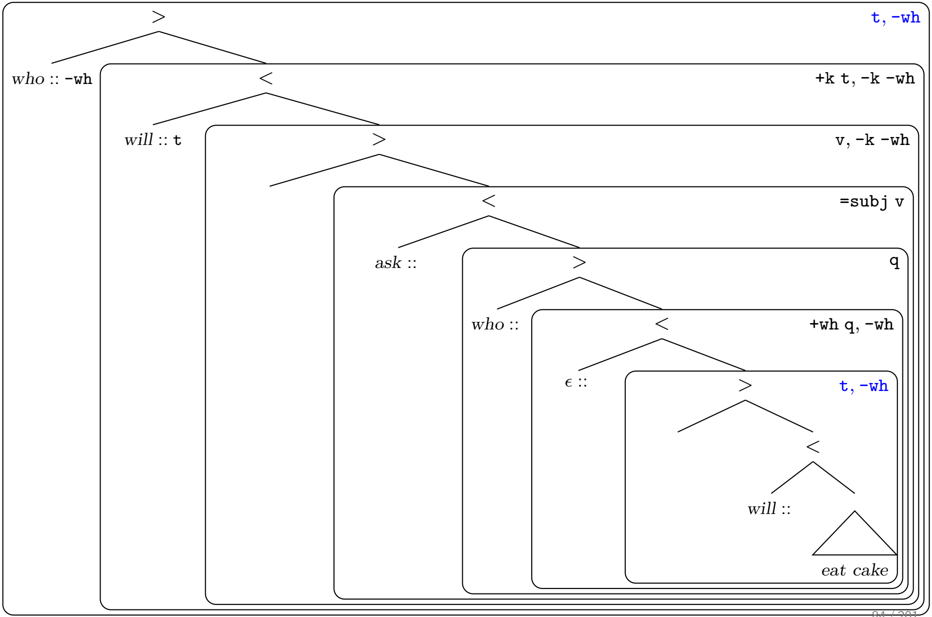
Extending with *who will ask . . .*



Extending with *who will ask ...*

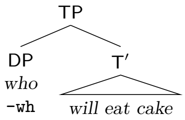


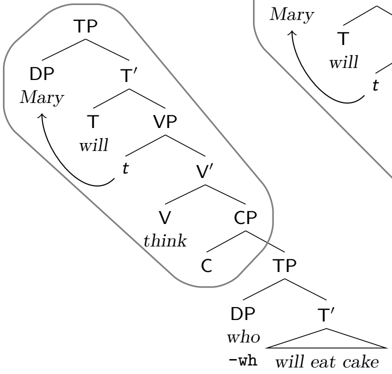
Extending with *who will ask* . . .

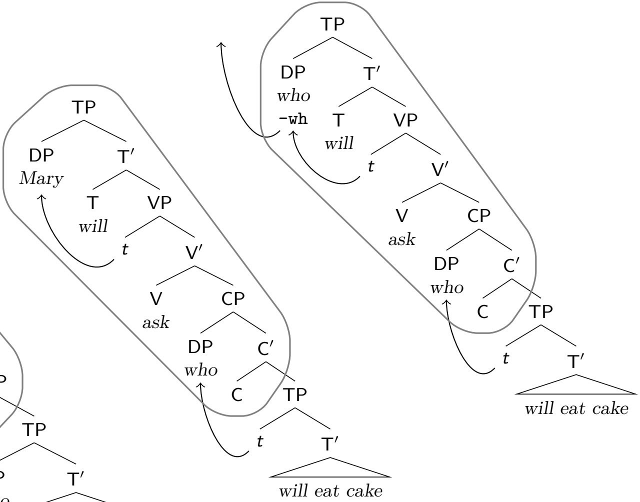


Which extensions create "loops"?

Starting point:



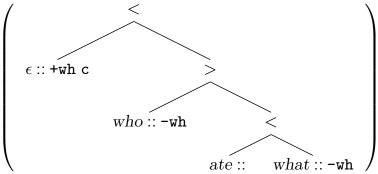




Importance of the SMC

The SMC ensures that there is a finite number of types (that we care about).

Recall: MOVE



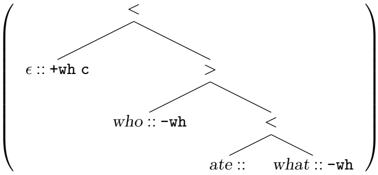
is **undefined**

(Michaelis 2001)

Importance of the SMC

The SMC ensures that there is a finite number of types (that we care about).

Recall: MOVE



is **undefined**

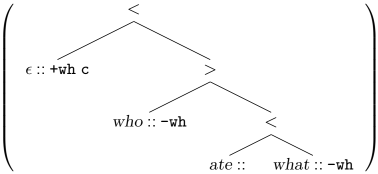
- So MOVE cannot be applied to expressions of type $\langle$+wh c, -wh, -wh$\rangle$.

(Michaelis 2001)

Importance of the SMC

The SMC ensures that there is a finite number of types (that we care about).

Recall: MOVE



is **undefined**

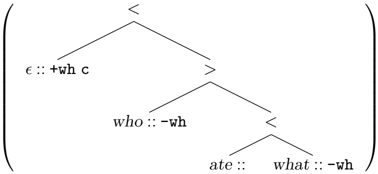
- So MOVE cannot be applied to expressions of type $\langle$+wh c, -wh, -wh$\rangle$.
- Nor to expressions of type $\langle$+wh c, -wh -k, -wh$\rangle$.
- These are “dead end” types.

(Michaelis 2001)

Importance of the SMC

The SMC ensures that there is a finite number of types (that we care about).

Recall: MOVE



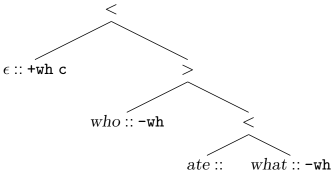
is **undefined**

- So MOVE cannot be applied to expressions of type $\langle$+wh c, -wh, -wh$\rangle$.
- Nor to expressions of type $\langle$+wh c, -wh -k, -wh$\rangle$.
- These are "dead end" types.
- An expression of type $\langle$t, -wh -k, -wh$\rangle$ can be the input to MERGE.

(Michaelis 2001)

Importance of the SMC

The SMC ensures that there is a finite number of types (that we care about).

Recall: MOVE (

) is **undefined**

- So MOVE cannot be applied to expressions of type ⟨+wh c, -wh, -wh⟩.
- Nor to expressions of type ⟨+wh c, -wh -k, -wh⟩.
- These are "dead end" types.

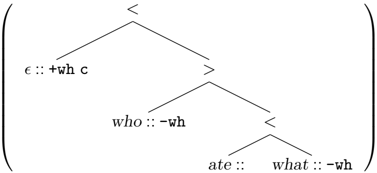
- An expression of type ⟨t, -wh -k, -wh⟩ can be the input to MERGE.
- But such types are also bound to lead to dead ends.

(Michaelis 2001)

Importance of the SMC

The SMC ensures that there is a finite number of types (that we care about).

Recall: MOVE



is **undefined**

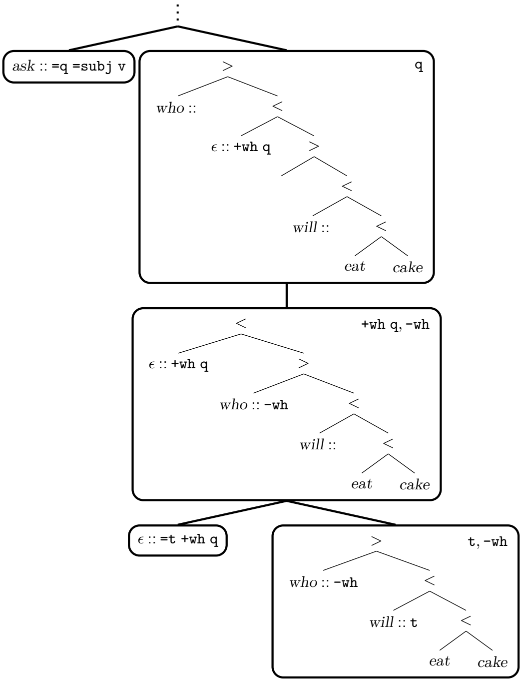
- So MOVE cannot be applied to expressions of type ⟨`+wh c`, `-wh`, `-wh`⟩.
- Nor to expressions of type ⟨`+wh c`, `-wh -k`, `-wh`⟩.
- These are "dead end" types.

- An expression of type ⟨`t`, `-wh -k`, `-wh`⟩ can be the input to MERGE.
- But such types are also bound to lead to dead ends.

So any type of the form $\langle \alpha, \ldots, \texttt{-f}\alpha_i, \ldots, \texttt{-f}\alpha_j, \ldots \rangle$ is not **useful**.
Thus there are only a finite number of useful types.

(Michaelis 2001)

Outline

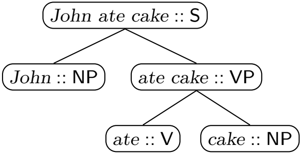

ask :: =q =subj v
q
who ::
ϵ :: +wh q
will ::
eat
cake
+wh q, -wh
ϵ :: +wh q
who :: -wh
will ::
eat
cake
ϵ :: =t +wh q
t, -wh
who :: -wh
will :: t
eat
cake

A possible concern

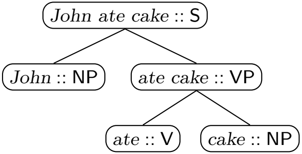
Question

"But hasn't our eventual derived expression lost the information that 'cake' is a DP?"

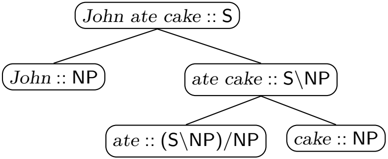
Derivations



Derivations



$$\dfrac{\mathit{John} :: \mathsf{NP} \qquad \dfrac{\mathit{ate} :: (\mathsf{S}\backslash\mathsf{NP})/\mathsf{NP} \qquad \mathit{cake} :: \mathsf{NP}}{\mathit{ate\ cake} :: \mathsf{S}\backslash\mathsf{NP}}}{\mathit{John\ ate\ cake} :: \mathsf{S}}$$



A possible concern

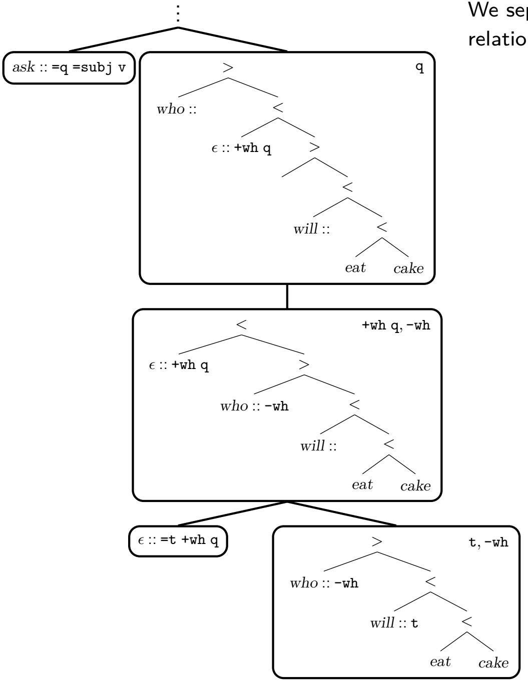
Question

"But hasn't our eventual derived expression lost the information that 'cake' is a DP?"

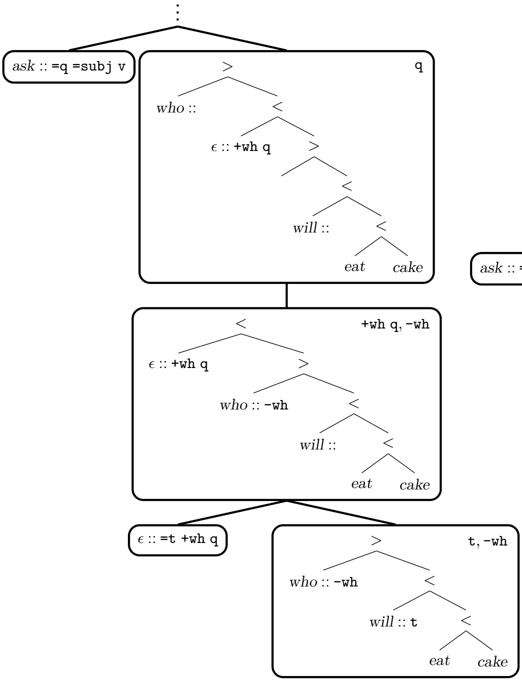
Answer

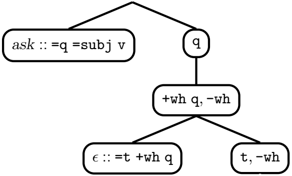
Yes, but only in the same way that $\textit{John ate cake} :: \mathsf{S}$ has also lost this information.

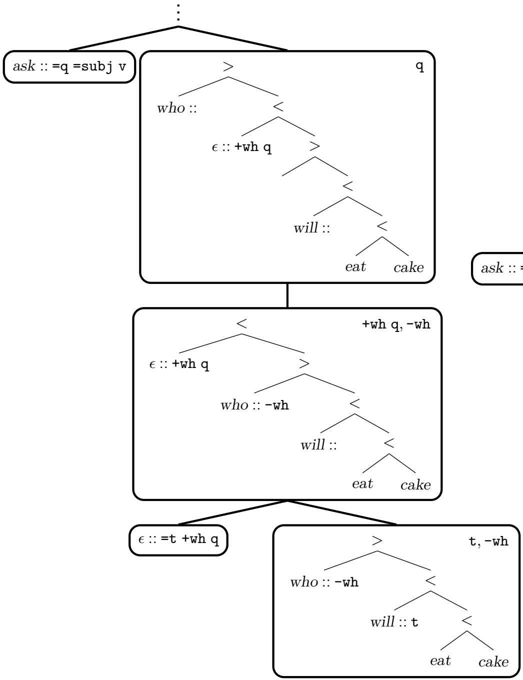
The point is not that we can look at the whole derivation to retrieve that information, the point is that the information has already done its job.

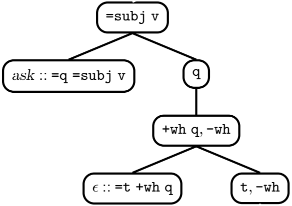


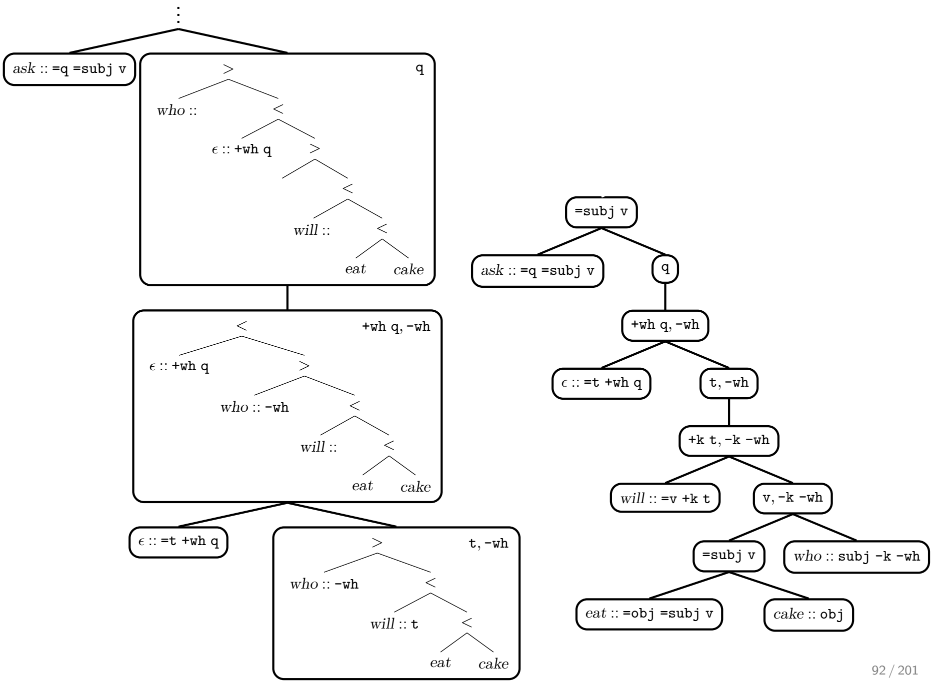
We separate the derivational precedence relation from the part-whole relation

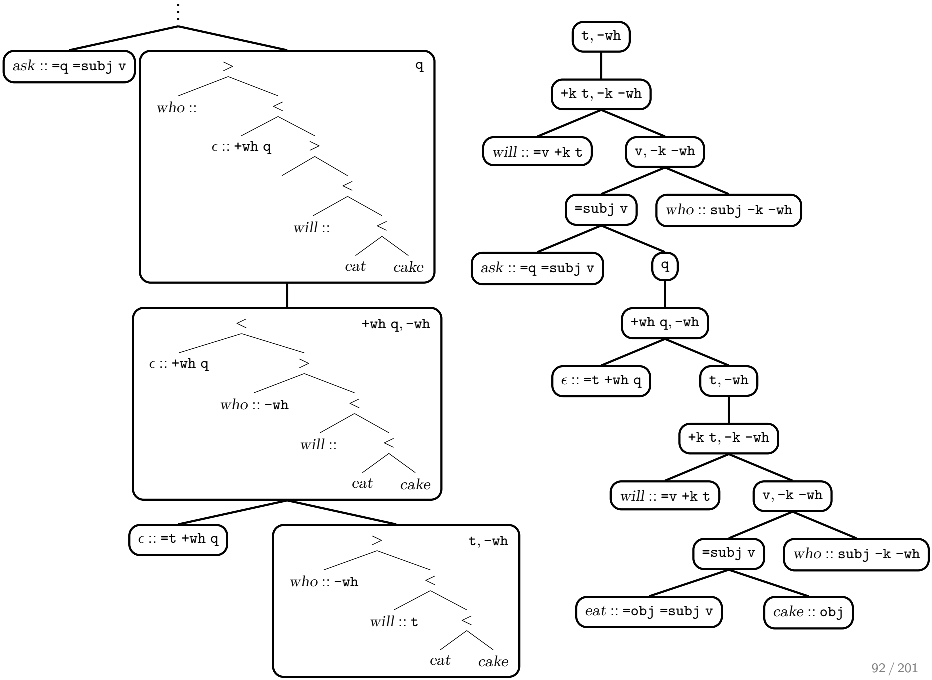

ask :: =q =subj v
>
q
who ::
<
ϵ :: +wh q
>
<
will ::
<
eat
cake
<
+wh q, -wh
ϵ :: +wh q
>
who :: -wh
<
will ::
<
eat
cake
ϵ :: =t +wh q
>
t, -wh
who :: -wh
<
will :: t
<
eat
cake

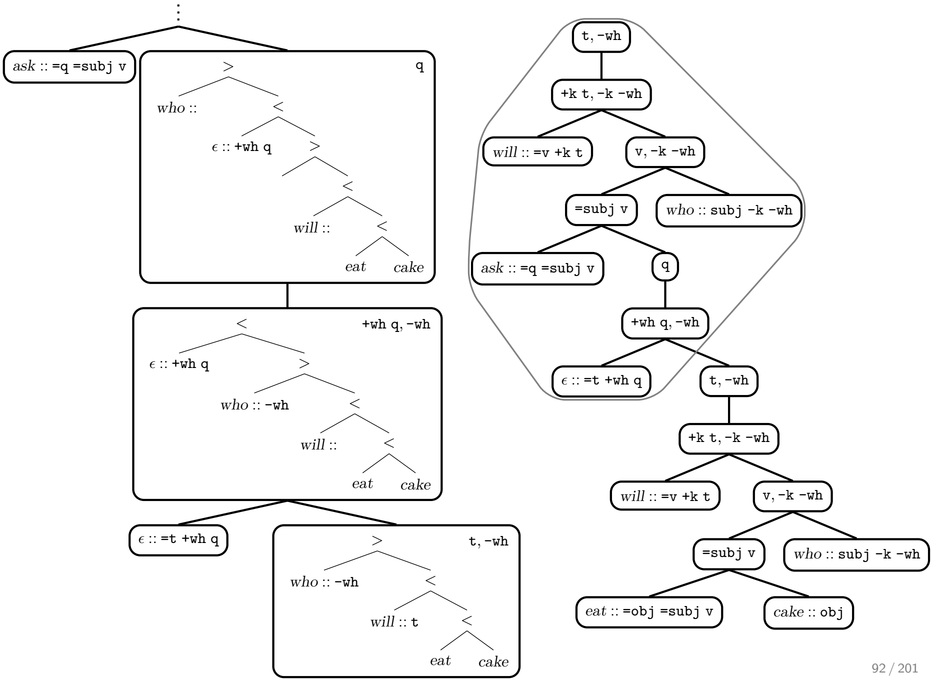

ask :: =q =subj v
q
+wh q, -wh
ϵ :: =t +wh q
t, -wh


ask :: =q =subj v
q
who ::
ϵ :: +wh q
will ::
eat cake
+wh q, -wh
ϵ :: +wh q
who :: -wh
will ::
eat cake
ϵ :: =t +wh q
t, -wh
who :: -wh
will :: t
eat cake

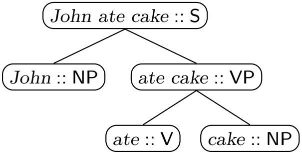

=subj v
ask :: =q =subj v
q
+wh q, -wh
ϵ :: =t +wh q
t, -wh


ask :: =q =subj v
q
who ::
ϵ :: +wh q
will ::
eat
cake
+wh q, -wh
ϵ :: +wh q
who :: -wh
will ::
eat
cake
ϵ :: =t +wh q
t, -wh
who :: -wh
will :: t
eat
cake
=subj v
ask :: =q =subj v
q
+wh q, -wh
ϵ :: =t +wh q
t, -wh
+k t, -k -wh
will :: =v +k t
v, -k -wh
=subj v
who :: subj -k -wh
eat :: =obj =subj v
cake :: obj

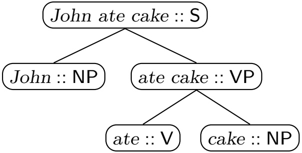

ask :: =q =subj v
>
q
who ::
<
ϵ :: +wh q
>
<
will ::
<
eat
cake
<
+wh q, -wh
ϵ :: +wh q
>
who :: -wh
<
will ::
<
eat
cake
ϵ :: =t +wh q
>
t, -wh
who :: -wh
<
will :: t
<
eat
cake
t, -wh
+k t, -k -wh
will :: =v +k t
v, -k -wh
=subj v
who :: subj -k -wh
ask :: =q =subj v
q
+wh q, -wh
ϵ :: =t +wh q
t, -wh
+k t, -k -wh
will :: =v +k t
v, -k -wh
=subj v
who :: subj -k -wh
eat :: =obj =subj v
cake :: obj

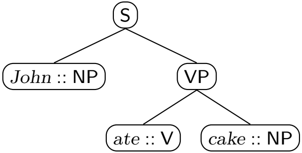

ask :: =q =subj v
q
who ::
ϵ :: +wh q
will ::
eat
cake
+wh q, -wh
ϵ :: +wh q
who :: -wh
will ::
eat
cake
ϵ :: =t +wh q
t, -wh
who :: -wh
will :: t
eat
cake
t, -wh
+k t, -k -wh
will :: =v +k t
v, -k -wh
=subj v
who :: subj -k -wh
ask :: =q =subj v
q
+wh q, -wh
ϵ :: =t +wh q
t, -wh
+k t, -k -wh
will :: =v +k t
v, -k -wh
=subj v
who :: subj -k -wh
eat :: =obj =subj v
cake :: obj

Labeling of internal nodes



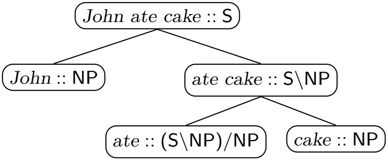
Labeling of internal nodes





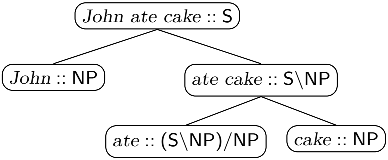
Labeling of internal nodes

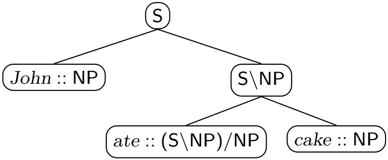
$$\dfrac{\mathit{John} :: \mathsf{NP} \qquad \dfrac{\mathit{ate} :: (\mathsf{S}\backslash\mathsf{NP})/\mathsf{NP} \qquad \mathit{cake} :: \mathsf{NP}}{\mathit{ate\ cake} :: \mathsf{S}\backslash\mathsf{NP}}}{\mathit{John\ ate\ cake} :: \mathsf{S}}$$

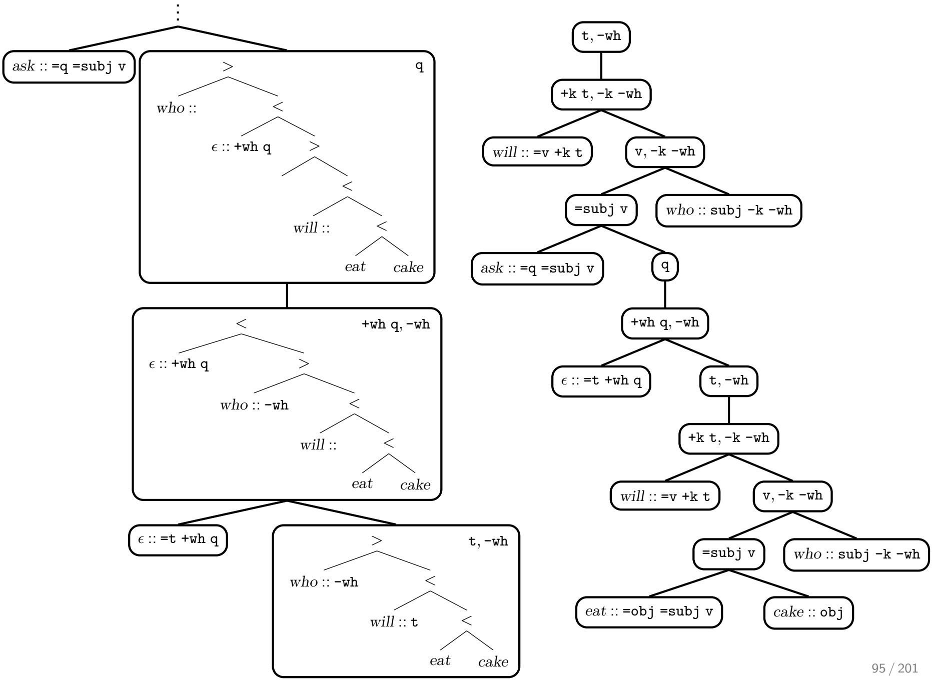


Labeling of internal nodes

$$\dfrac{\textit{John} :: \mathsf{NP} \qquad \dfrac{\textit{ate} :: (\mathsf{S}\backslash\mathsf{NP})/\mathsf{NP} \qquad \textit{cake} :: \mathsf{NP}}{\textit{ate cake} :: \mathsf{S}\backslash\mathsf{NP}}}{\textit{John ate cake} :: \mathsf{S}}$$






ask :: =q =subj v
q
who ::
ϵ :: +wh q
will ::
eat
cake
+wh q, -wh
ϵ :: +wh q
who :: -wh
will ::
eat
cake
ϵ :: =t +wh q
t, -wh
who :: -wh
will :: t
eat
cake
t, -wh
+k t, -k -wh
will :: =v +k t
v, -k -wh
=subj v
who :: subj -k -wh
ask :: =q =subj v
q
+wh q, -wh
ϵ :: =t +wh q
t, -wh
+k t, -k -wh
will :: =v +k t
v, -k -wh
=subj v
who :: subj -k -wh
eat :: =obj =subj v
cake :: obj

Context-free structure

$$\begin{aligned}
\langle \texttt{=subj v} \rangle &\rightarrow \langle \texttt{=q =subj v} \rangle \quad \langle \texttt{q} \rangle \\
\langle \texttt{q} \rangle &\rightarrow \langle \texttt{+wh q}, \texttt{-wh} \rangle \\
\langle \texttt{+wh q}, \texttt{-wh} \rangle &\rightarrow \langle \texttt{=t +wh q} \rangle \quad \langle \texttt{t}, \texttt{-wh} \rangle
\end{aligned}$$

Context-free structure

$$\begin{aligned}
\langle \texttt{=subj v} \rangle &\rightarrow \langle \texttt{=q =subj v} \rangle \quad \langle \texttt{q} \rangle \\
\langle \texttt{q} \rangle &\rightarrow \langle \texttt{+wh q}, \texttt{-wh} \rangle \\
\langle \texttt{+wh q}, \texttt{-wh} \rangle &\rightarrow \langle \texttt{=t +wh q} \rangle \quad \langle \texttt{t}, \texttt{-wh} \rangle
\end{aligned}$$

General schemas for MERGE steps (approximate):

$$\begin{aligned}
\langle \gamma, \alpha_1, \ldots, \alpha_j, \beta_1, \ldots, \beta_k \rangle &\rightarrow \langle \texttt{=f}\gamma, \alpha_1, \ldots, \alpha_j \rangle \quad \langle \texttt{f}, \beta_1, \ldots, \beta_k \rangle \\
\langle \gamma, \alpha_1, \ldots, \alpha_j, \delta, \beta_1, \ldots, \beta_k \rangle &\rightarrow \langle \texttt{=f}\gamma, \alpha_1, \ldots, \alpha_j \rangle \quad \langle \texttt{f}\delta, \beta_1, \ldots, \beta_k \rangle
\end{aligned}$$

General schemas for MOVE steps (approximate):

$$\begin{aligned}
\langle \gamma, \alpha_1, \ldots, \alpha_{i-1}, \alpha_{i+1}, \ldots, \alpha_k \rangle &\rightarrow \langle \texttt{+f}\gamma, \alpha_1, \ldots, \alpha_{i-1}, \texttt{-f}, \alpha_{i+1}, \ldots, \alpha_k \rangle \\
\langle \gamma, \alpha_1, \ldots, \alpha_{i-1}, \delta, \alpha_{i+1}, \ldots, \alpha_k \rangle &\rightarrow \langle \texttt{+f}\gamma, \alpha_1, \ldots, \alpha_{i-1}, \texttt{-f}\delta, \alpha_{i+1}, \ldots, \alpha_k \rangle
\end{aligned}$$

Context-free structure

$$
\begin{array}{rcll}
\langle \texttt{=subj v} \rangle & \rightarrow & \langle \texttt{=q =subj v} \rangle & \langle \texttt{q} \rangle \\
\langle \texttt{q} \rangle & \rightarrow & \langle \texttt{+wh q}, \texttt{-wh} \rangle & \\
\langle \texttt{+wh q}, \texttt{-wh} \rangle & \rightarrow & \langle \texttt{=t +wh q} \rangle & \langle \texttt{t}, \texttt{-wh} \rangle
\end{array}
$$

General schemas for MERGE steps (approximate):

$$
\begin{array}{rcll}
\langle \gamma, \alpha_1, \dots, \alpha_j, \beta_1, \dots, \beta_k \rangle & \rightarrow & \langle \texttt{=f}\gamma, \alpha_1, \dots, \alpha_j \rangle & \langle \texttt{f}, \beta_1, \dots, \beta_k \rangle \\
\langle \gamma, \alpha_1, \dots, \alpha_j, \delta, \beta_1, \dots, \beta_k \rangle & \rightarrow & \langle \texttt{=f}\gamma, \alpha_1, \dots, \alpha_j \rangle & \langle \texttt{f}\delta, \beta_1, \dots, \beta_k \rangle
\end{array}
$$

General schemas for MOVE steps (approximate):

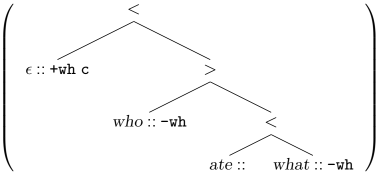
$$
\begin{array}{rcl}
\langle \gamma, \alpha_1, \dots, \alpha_{i-1}, \alpha_{i+1}, \dots, \alpha_k \rangle & \rightarrow & \langle \texttt{+f}\gamma, \alpha_1, \dots, \alpha_{i-1}, \texttt{-f}, \alpha_{i+1}, \dots, \alpha_k \rangle \\
\langle \gamma, \alpha_1, \dots, \alpha_{i-1}, \delta, \alpha_{i+1}, \dots, \alpha_k \rangle & \rightarrow & \langle \texttt{+f}\gamma, \alpha_1, \dots, \alpha_{i-1}, \texttt{-f}\delta, \alpha_{i+1}, \dots, \alpha_k \rangle
\end{array}
$$

- MOVE steps **change** something without **combining** it with anything
- Compare with unary CFG rules, or type-raising in CCG, or ...

Importance of the SMC

The SMC ensures that there is a finite number of types (that we care about).

Recall: MOVE



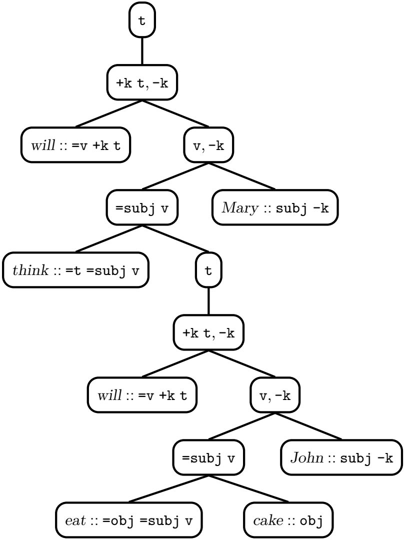
is **undefined**

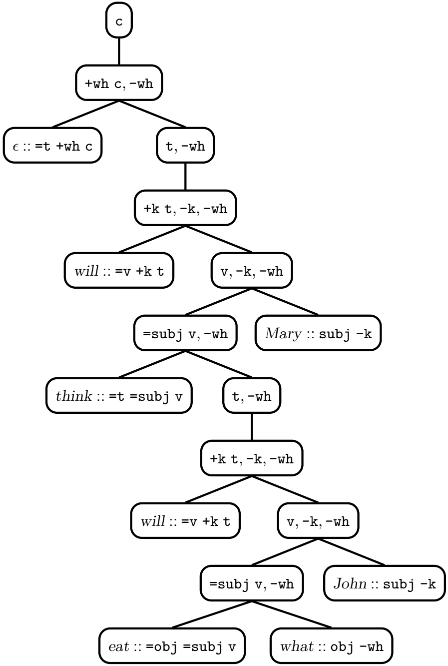
- So MOVE cannot be applied to expressions of type ⟨+wh c, -wh, -wh⟩.
- Nor to expressions of type ⟨+wh c, -wh -k, -wh⟩.
- These are "dead end" types.

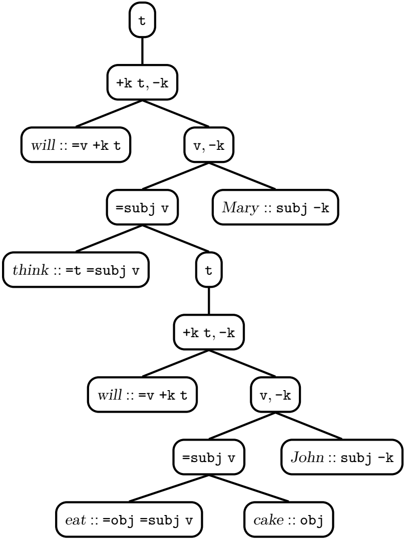
- An expression of type ⟨t, -wh -k, -wh⟩ can be the input to MERGE.
- But such types are also bound to lead to dead ends.

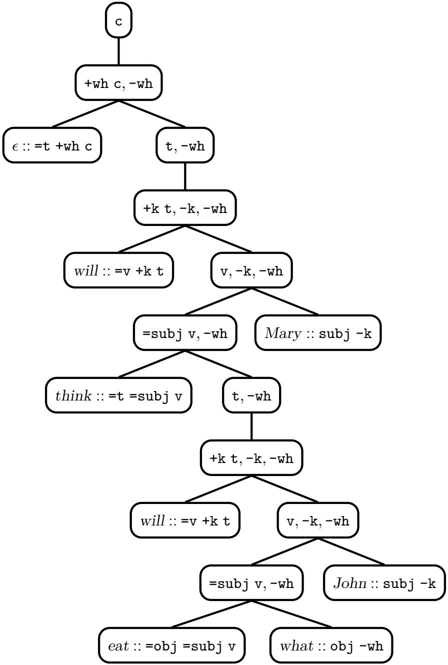
So any type of the form $\langle \alpha, \ldots, \texttt{-f}\alpha_i, \ldots, \texttt{-f}\alpha_j, \ldots \rangle$ is not **useful**.
Thus there are only a finite number of useful types.

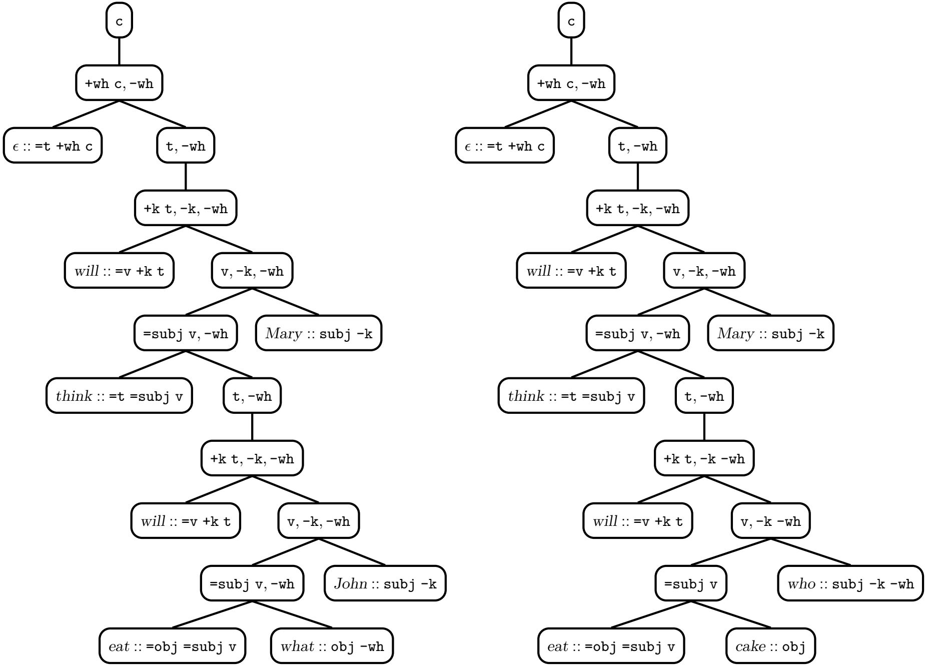
(Michaelis 2001)


t
+k t, -k
will :: =v +k t
v, -k
=subj v
Mary :: subj -k
think :: =t =subj v
t
+k t, -k
will :: =v +k t
v, -k
=subj v
John :: subj -k
eat :: =obj =subj v
cake :: obj


c
+wh c, -wh
ϵ :: =t +wh c
t, -wh
+k t, -k, -wh
will :: =v +k t
v, -k, -wh
=subj v, -wh
Mary :: subj -k
think :: =t =subj v
t, -wh
+k t, -k, -wh
will :: =v +k t
v, -k, -wh
=subj v, -wh
John :: subj -k
eat :: =obj =subj v
what :: obj -wh


t
+k t, -k
will :: =v +k t
v, -k
=subj v
Mary :: subj -k
think :: =t =subj v
t
+k t, -k
will :: =v +k t
v, -k
=subj v
John :: subj -k
eat :: =obj =subj v
cake :: obj


c
+wh c, -wh
ϵ :: =t +wh c
t, -wh
+k t, -k, -wh
will :: =v +k t
v, -k, -wh
=subj v, -wh
Mary :: subj -k
think :: =t =subj v
t, -wh
+k t, -k, -wh
will :: =v +k t
v, -k, -wh
=subj v, -wh
John :: subj -k
eat :: =obj =subj v
what :: obj -wh


c
+wh c, -wh
ϵ :: =t +wh c
t, -wh
+k t, -k, -wh
will :: =v +k t
v, -k, -wh
=subj v, -wh
Mary :: subj -k
think :: =t =subj v
t, -wh
+k t, -k, -wh
will :: =v +k t
v, -k, -wh
=subj v, -wh
John :: subj -k
eat :: =obj =subj v
what :: obj -wh
c
+wh c, -wh
ϵ :: =t +wh c
t, -wh
+k t, -k, -wh
will :: =v +k t
v, -k, -wh
=subj v, -wh
Mary :: subj -k
think :: =t =subj v
t, -wh
+k t, -k -wh
will :: =v +k t
v, -k -wh
=subj v
who :: subj -k -wh
eat :: =obj =subj v
cake :: obj

Sharpening the empirical claims of generative syntax through formalization

Tim Hunter — ESSLLI, August 2015

Part 3

MGs and MCFGs

Where we're up to

We've seen:

- MGs with operations defined that manipulated trees
- that the structure that "really matters" (e.g. for recursion) can be boiled down to funny-looking "derivation trees" (with things like $\langle \texttt{t}, \texttt{-k} \rangle$ at the non-leaf nodes)

Now:

- A way to think of how these derivation trees relate to surface strings (without going via trees)
- In some ways not totally necessary for the rest of the course, but helpful

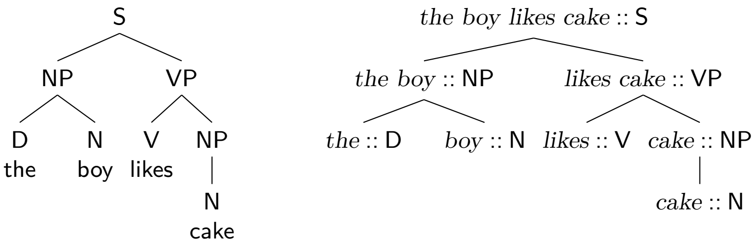
Later:

- Adding probabilities to MGs: in a way that sort of works, and does some good stuff, but doesn't do everything we'd want
- Adding probabilities to MGs: in an even better way

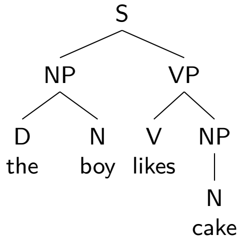
Outline

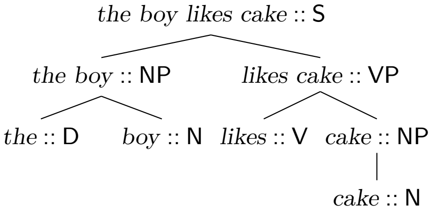
Outline

Trees


S
NP
VP
D
the
N
boy
V
likes
NP
N
cake
the boy likes cake :: S
the boy :: NP
likes cake :: VP
the :: D
boy :: N
likes :: V
cake :: NP
cake :: N

Trees





How to think of a tree:

- less as a picture of a string
- more as a graphical representation of how a string was constructed, with the string "at" the top node

Two sides of a CFG rule

A rule like ‘S $\rightarrow$ NP VP’ says two things:

- What combines with what:
  An NP and a VP can combine to form an S
- How to produce a string of the new category:
  Put the NP-string to the left of the VP-string

More explicitly:

$$st :: \mathrm{S} \quad \rightarrow \quad s :: \mathrm{NP} \quad t :: \mathrm{VP}$$

Example: X-bar theory

Japanese

$\text{XP} \rightarrow \text{Spec X}'$

$\text{X}' \rightarrow \text{Comp X}$

English

$\text{XP} \rightarrow \text{Spec X}'$

$\text{X}' \rightarrow \text{X Comp}$

Example: X-bar theory

Japanese

$\text{XP} \rightarrow \text{Spec X}'$

$\text{X}' \rightarrow \text{Comp X}$

English

$\text{XP} \rightarrow \text{Spec X}'$

$\text{X}' \rightarrow \text{X Comp}$

Japanese

$$st :: \text{XP} \rightarrow s :: \text{Spec} \quad t :: \text{X}'$$

$$st :: \text{X}' \rightarrow s :: \text{Comp} \quad t :: \text{X}$$

English

$$st :: \text{XP} \rightarrow s :: \text{Spec} \quad t :: \text{X}'$$

$$ts :: \text{X}' \rightarrow s :: \text{Comp} \quad t :: \text{X}$$

Example: X-bar theory

Japanese

XP → Spec X′
X′ → Comp X

English

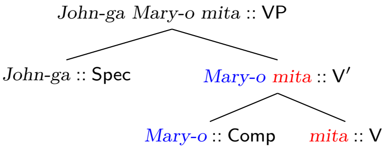
XP → Spec X′
X′ → X Comp

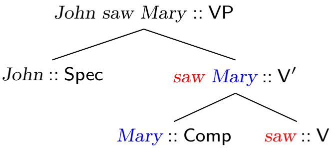
Japanese

$$st :: \text{XP} \rightarrow s :: \text{Spec} \quad t :: \text{X}'$$
$$st :: \text{X}' \rightarrow s :: \text{Comp} \quad t :: \text{X}$$

English

$$st :: \text{XP} \rightarrow s :: \text{Spec} \quad t :: \text{X}'$$
$$ts :: \text{X}' \rightarrow s :: \text{Comp} \quad t :: \text{X}$$





Outline

Concatenative and non-concatenative operations

Concatenative morphology:

play + ed	$\rightsquigarrow$	played
play + ing	$\rightsquigarrow$	playing
play + s	$\rightsquigarrow$	plays

Non-concatenative morphology:

(k,t,b) + (i,aa)	$\rightsquigarrow$	kitaab	("book")
(k,t,b) + (aa,i)	$\rightsquigarrow$	kaatib	("writer")
(k,t,b) + (ma,uu)	$\rightsquigarrow$	maktuub	("written")
(k,t,b) + (a,i,a)	$\rightsquigarrow$	katiba	("document")

Concatenative and non-concatenative operations

Concatenative morphology:

play + ed	$\rightsquigarrow$	played
play + ing	$\rightsquigarrow$	playing
play + s	$\rightsquigarrow$	plays

Non-concatenative morphology:

(k,t,b) + (i,aa)	$\rightsquigarrow$	kitaab	("book")
(k,t,b) + (aa,i)	$\rightsquigarrow$	kaatib	("writer")
(k,t,b) + (ma,uu)	$\rightsquigarrow$	maktuub	("written")
(k,t,b) + (a,i,a)	$\rightsquigarrow$	katiba	("document")

Concatenative syntax:

plays + tennis	$\rightsquigarrow$	plays tennis
plays + soccer	$\rightsquigarrow$	plays soccer
John + plays soccer	$\rightsquigarrow$	John plays soccer
Mary + plays soccer	$\rightsquigarrow$	Mary plays soccer

Concatenative and non-concatenative operations

Concatenative morphology:

play + ed	$\rightsquigarrow$	played
play + ing	$\rightsquigarrow$	playing
play + s	$\rightsquigarrow$	plays

Non-concatenative morphology:

(k,t,b) + (i,aa)	$\rightsquigarrow$	kitaab	("book")
(k,t,b) + (aa,i)	$\rightsquigarrow$	kaatib	("writer")
(k,t,b) + (ma,uu)	$\rightsquigarrow$	maktuub	("written")
(k,t,b) + (a,i,a)	$\rightsquigarrow$	katiba	("document")

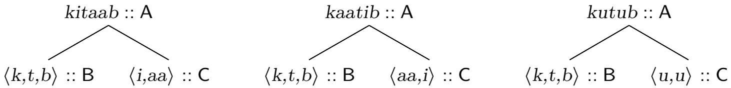
Concatenative syntax:

plays + tennis	$\rightsquigarrow$	plays tennis
plays + soccer	$\rightsquigarrow$	plays soccer
John + plays soccer	$\rightsquigarrow$	John plays soccer
Mary + plays soccer	$\rightsquigarrow$	Mary plays soccer

Non-concatenative syntax:

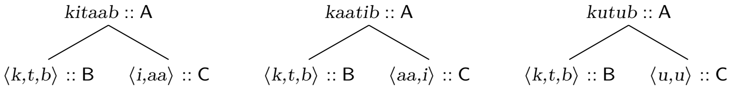
seems + (John, to be tall)	$\rightsquigarrow$	John seems to be tall
seems + (Mary, to be intelligent)	$\rightsquigarrow$	Mary seems to be intelligent
did + (John see, who)	$\rightsquigarrow$	who did John see
did + (Mary meet, who)	$\rightsquigarrow$	who did Mary meet

Non-concatenative morphology

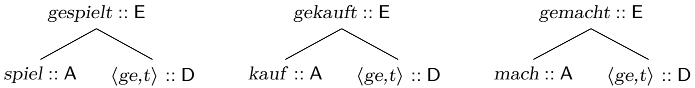


$$stuvw :: \mathsf{A} \quad \rightarrow \quad \langle s, u, w \rangle :: \mathsf{B} \quad \langle t, v \rangle :: \mathsf{C}$$

Non-concatenative morphology



$$stuvw :: \mathsf{A} \quad \rightarrow \quad \langle s, u, w \rangle :: \mathsf{B} \quad \langle t, v \rangle :: \mathsf{C}$$



$$stu :: \mathsf{E} \quad \rightarrow \quad t :: \mathsf{A} \quad \langle s, u \rangle :: \mathsf{D}$$

Non-concatenative morphology

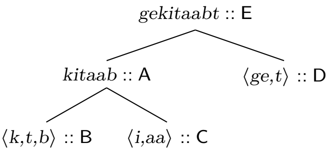
$$stuvw :: \mathrm{A} \quad \rightarrow \quad \langle s, u, w \rangle :: \mathrm{B} \quad \langle t, v \rangle :: \mathrm{C}$$

$$stu :: \mathrm{E} \quad \rightarrow \quad t :: \mathrm{A} \quad \langle s, u \rangle :: \mathrm{D}$$

Non-concatenative morphology

$$stuvw :: \mathsf{A} \quad \rightarrow \quad \langle s, u, w \rangle :: \mathsf{B} \quad \langle t, v \rangle :: \mathsf{C}$$

$$stu :: \mathsf{E} \quad \rightarrow \quad t :: \mathsf{A} \quad \langle s, u \rangle :: \mathsf{D}$$

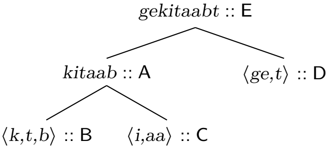


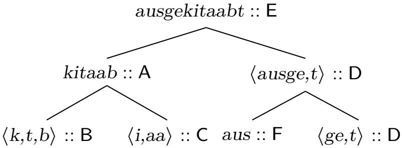
Non-concatenative morphology

$$stuvw :: \mathsf{A} \rightarrow \langle s, u, w\rangle :: \mathsf{B} \quad \langle t, v\rangle :: \mathsf{C}$$

$$stu :: \mathsf{E} \rightarrow t :: \mathsf{A} \quad \langle s, u\rangle :: \mathsf{D}$$

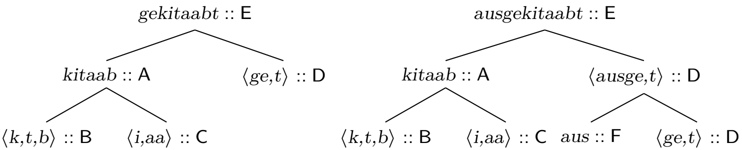
$$\langle ts, u\rangle :: \mathsf{D} \rightarrow t :: \mathsf{F} \quad \langle s, u\rangle :: \mathsf{D}$$





Non-concatenative morphology

$$
\begin{aligned}
stuvw :: \mathsf{A} &\rightarrow \langle s, u, w\rangle :: \mathsf{B} \quad \langle t, v\rangle :: \mathsf{C} \\
stu :: \mathsf{E} &\rightarrow t :: \mathsf{A} \quad \langle s, u\rangle :: \mathsf{D} \\
\langle ts, u\rangle :: \mathsf{D} &\rightarrow t :: \mathsf{F} \quad \langle s, u\rangle :: \mathsf{D}
\end{aligned}
$$



If our goal is to characterize the array of well-formed/derivable objects — not to pronounce them — then all we care about is "what's built out of what":

$$
\begin{aligned}
\mathsf{A} &\rightarrow \mathsf{B} \quad \mathsf{C} \\
\mathsf{E} &\rightarrow \mathsf{A} \quad \mathsf{D} \\
\mathsf{D} &\rightarrow \mathsf{F} \quad \mathsf{D}
\end{aligned}
$$

Outline

Multiple Context-Free Grammars (MCFGs)

$$st :: \mathrm{S} \quad \rightarrow \quad s :: \mathrm{NP} \quad t :: \mathrm{VP}$$

An MCFG generalises to allow yields to be *tuples of strings*.

$$t_2 s t_1 :: \mathrm{Q} \quad \rightarrow \quad s :: \mathrm{NP} \quad \langle t_1, t_2 \rangle :: \mathrm{VPWH}$$

This rule says two things:

- We can combine an NP with a VPWH to make a Q.
- The yield of the Q is $t_2 s t_1$,
  where s is the yield of the NP and $\langle t_1, t_2 \rangle$ is the yield of the VPWH.

Multiple Context-Free Grammars (MCFGs)

$$st :: \text{S} \quad \rightarrow \quad s :: \text{NP} \quad t :: \text{VP}$$

An MCFG generalises to allow yields to be *tuples of strings*.

$$t_2 s t_1 :: \text{Q} \quad \rightarrow \quad s :: \text{NP} \quad \langle t_1, t_2 \rangle :: \text{VPWH}$$

This rule says two things:

- We can combine an NP with a VPWH to make a Q.
- The yield of the Q is $t_2 s t_1$,
  where s is the yield of the NP and $\langle t_1, t_2 \rangle$ is the yield of the VPWH.

$$\textit{which girl the boy says is tall} :: \text{Q} \quad \rightarrow$$
$$\textit{the boy} :: \text{NP} \quad \langle \textit{says is tall}, \textit{which girl} \rangle :: \text{VPWH}$$

Some technical details

- Each nonterminal has a rank n, and yields only n-tuples of strings.

  So given this rule:

  $$t_2 s t_1 :: \mathrm{Q} \quad \rightarrow \quad s :: \mathrm{NP} \quad \langle t_1, t_2 \rangle :: \mathrm{VPWH}$$

  we know that anything producing a VPWH must produce a 2-tuple.

  $$\langle \ldots, \ldots \rangle :: \mathrm{VPWH} \quad \rightarrow \quad \ldots$$

  and that anything producing an NP must produce a 1-tuple:

  $$\ldots :: \mathrm{NP} \quad \rightarrow \quad \ldots$$

(Seki et al. 1991, Weir 1988, Vijay-Shanker et al. 1987)

Some technical details

- Each nonterminal has a rank n, and yields only n-tuples of strings.

  So given this rule:

  $$t_2 s t_1 :: \text{Q} \quad \rightarrow \quad s :: \text{NP} \quad \langle t_1, t_2 \rangle :: \text{VPWH}$$

  we know that anything producing a VPWH must produce a 2-tuple.

  $$\langle \ldots, \ldots \rangle :: \text{VPWH} \quad \rightarrow \quad \ldots$$

  and that anything producing an NP must produce a 1-tuple:

  $$\ldots :: \text{NP} \quad \rightarrow \quad \ldots$$

- The string-composition functions cannot copy pieces of their arguments.

OK	$s\,t :: \text{VP}$	$\rightarrow$	$s :: \text{V}$	$t :: \text{NP}$
OK	$t\,s\,\textit{himself} :: \text{S}$	$\rightarrow$	$s :: \text{V}$	$t :: \text{NP}$
Not OK	$t\,s\,t :: \text{S}$	$\rightarrow$	$s :: \text{V}$	$t :: \text{NP}$

(Seki et al. 1991, Weir 1988, Vijay-Shanker et al. 1987)

Some technical details

- Each nonterminal has a rank n, and yields only n-tuples of strings.

  So given this rule:

  $$t_2 s t_1 :: \text{Q} \quad \rightarrow \quad s :: \text{NP} \quad \langle t_1, t_2 \rangle :: \text{VPWH}$$

  we know that anything producing a VPWH must produce a 2-tuple.

  $$\langle \ldots, \ldots \rangle :: \text{VPWH} \quad \rightarrow \quad \ldots$$

  and that anything producing an NP must produce a 1-tuple:

  $$\ldots :: \text{NP} \quad \rightarrow \quad \ldots$$

- The string-composition functions cannot copy pieces of their arguments.

OK	$s\,t :: \text{VP}$	$\rightarrow$	$s :: \text{V}$	$t :: \text{NP}$
OK	$t\,s\,\textit{himself} :: \text{S}$	$\rightarrow$	$s :: \text{V}$	$t :: \text{NP}$
Not OK	$t\,s\,t :: \text{S}$	$\rightarrow$	$s :: \text{V}$	$t :: \text{NP}$

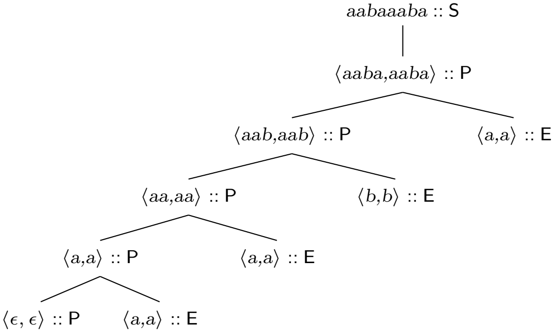
- Essentially equivalent to linear context-free rewriting systems (LCFRSs).

(Seki et al. 1991, Weir 1988, Vijay-Shanker et al. 1987)

Beyond context-free

$$
\begin{aligned}
t_1 t_2 :: \mathsf{S} &\quad\rightarrow\quad \langle t_1, t_2 \rangle :: \mathsf{P} \\
\langle t_1 u_1, t_2 u_2 \rangle :: \mathsf{P} &\quad\rightarrow\quad \langle t_1, t_2 \rangle :: \mathsf{P} \quad \langle u_1, u_2 \rangle :: \mathsf{E} \\
\langle \epsilon, \epsilon \rangle :: \mathsf{P} & \\
\langle a, a \rangle :: \mathsf{E} & \\
\langle b, b \rangle :: \mathsf{E} &
\end{aligned}
$$

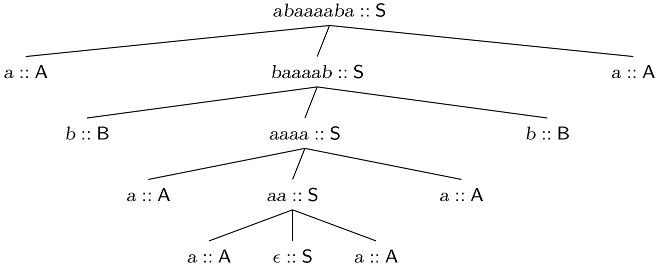
$$\left\{ ww \mid w \in \{a, b\}^* \right\}$$



Unlike in a CFG, we can ensure that the two "halves" are extended in the same ways without concatenating them together.

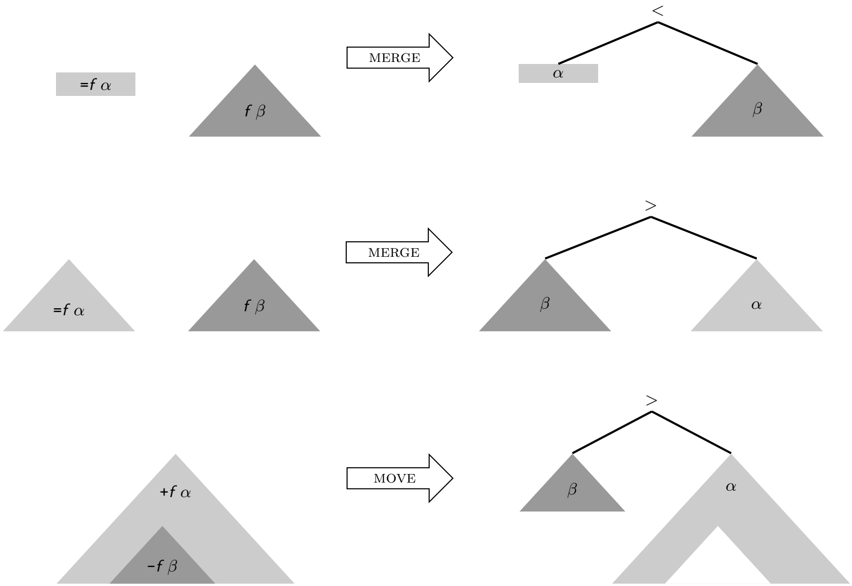
For comparison

$$
\begin{aligned}
t_1 s t_2 :: \mathsf{S} &\rightarrow t_1 :: \mathsf{A} \quad s :: \mathsf{S} \quad t_2 :: \mathsf{A} \\
t_1 s t_2 :: \mathsf{S} &\rightarrow t_1 :: \mathsf{B} \quad s :: \mathsf{S} \quad t_2 :: \mathsf{B} \\
\epsilon :: \mathsf{S} & \\
a :: \mathsf{A} & \\
b :: \mathsf{B} &
\end{aligned}
$$



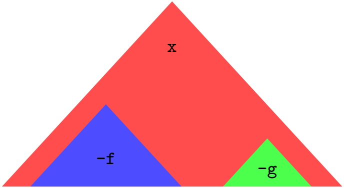
Outline

Merge and move



What matters in a (derived) tree

This tree:



becomes a tuple of categorized strings:

$$\left\langle\; s :: \mathtt{x} \;,\; t :: \mathtt{-f} \;,\; u :: \mathtt{-g} \;\right\rangle_0$$

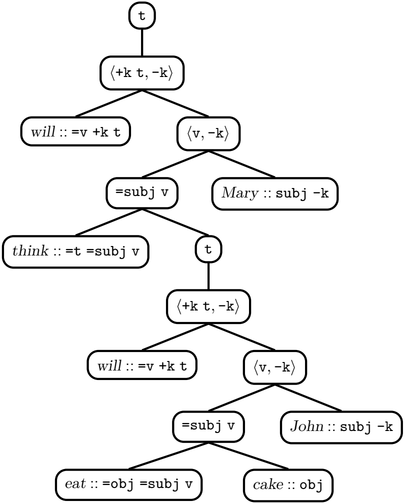
or, equivalently, a tuple-of-strings, categorized by a tuple-of-categories:

$$\langle s, t, u\rangle :: \langle \mathtt{x}, \mathtt{-f}, \mathtt{-g}\rangle_0$$

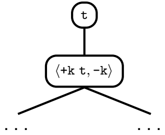
(Michaelis 2001, Stabler and Keenan 2003)

Remember MG derivation trees?

- We can tell that this tree represents a well-formed derivation, by checking the feature-manipulations at each step.
- How can we work out which string it derives?
  - Build up a tree according to merge and move rules, and read off leaves of the tree.
  - But there's a simpler way.



Producing a string from a derivation tree

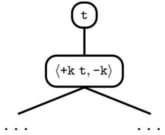


What do we need to have computed at the ⟨+k t, -k⟩ node, in order to compute the final string

Mary will think John will eat cake

at the t node?

Producing a string from a derivation tree

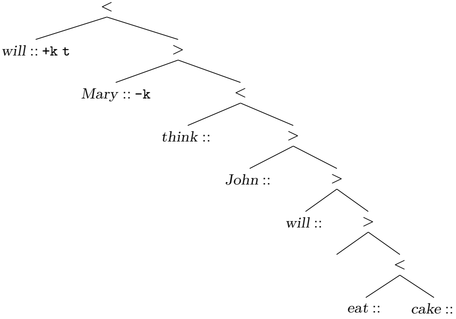


What do we need to have computed at the ⟨+k t, -k⟩ node, in order to compute the final string

Mary will think John will eat cake

at the t node?

This tree would do:

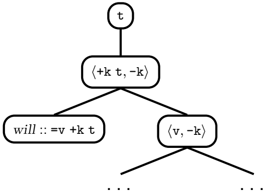


But all we actually need to know is:

- What's the string corresponding to the part that's going to move to check -k?
- What's the string corresponding to the leftovers?

These questions are answered by the tuple ⟨*will think John will eat cake*, *Mary*⟩

Producing a string from a derivation tree

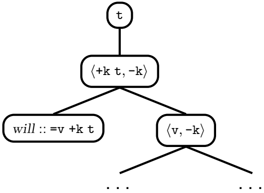


What do we need to have computed at the $\langle \texttt{v}, \texttt{-k} \rangle$ node, in order to compute the desired tuple

$$\langle \textit{will think John will eat cake, Mary} \rangle$$

at the $\langle \texttt{+k t}, \texttt{-k} \rangle$ node?

Producing a string from a derivation tree

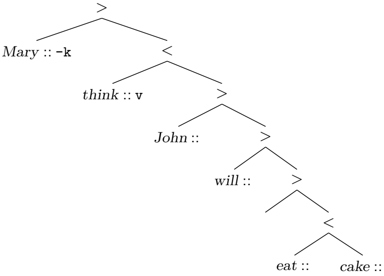


What do we need to have computed at the ⟨v, -k⟩ node, in order to compute the desired tuple

$$\langle \textit{will think John will eat cake}, \textit{Mary} \rangle$$

at the ⟨+k t, -k⟩ node?

This tree would do:

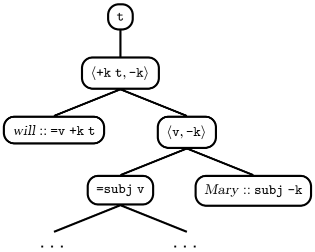


But all we actually need to know is:

- What's the string corresponding to the part that's going to move to check -k?
- What's the string corresponding to the leftovers?

These questions are answered by the tuple $\langle \textit{think John will eat cake}, \textit{Mary} \rangle$

Producing a string from a derivation tree

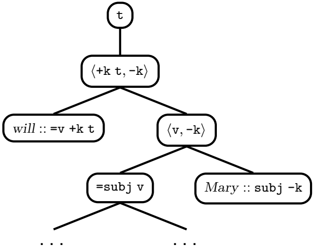


What do we need to have computed at the `=subj v` node, in order to compute the desired tuple

$$\langle \textit{think John will eat cake, Mary} \rangle$$

at the ⟨`v`, `-k`⟩ node?

Producing a string from a derivation tree

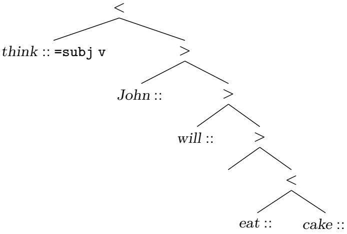


What do we need to have computed at the `=subj v` node, in order to compute the desired tuple

$$\langle \textit{think John will eat cake, Mary} \rangle$$

at the ⟨`v`, `-k`⟩ node?

This tree would do:



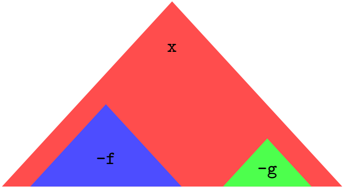
But all we actually need to know is:

- What's the string corresponding to the entire tree? (The "leftovers after no movement".)

This question is answered by the string *think John will eat cake*

What matters in a (derived) tree

This tree:



becomes a tuple of categorized strings:

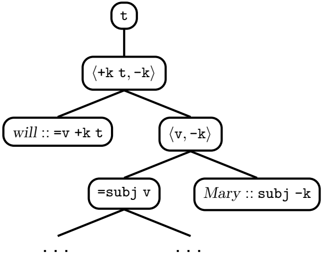
$$\left\langle \; s :: \mathtt{x} \; , \; t :: \mathtt{-f} \; , \; u :: \mathtt{-g} \; \right\rangle_0$$

or, equivalently, a tuple-of-strings, categorized by a tuple-of-categories:

$$\langle s, t, u \rangle :: \langle \mathtt{x}, \mathtt{-f}, \mathtt{-g} \rangle_0$$

(Michaelis 2001, Stabler and Keenan 2003)

MCFG rules



$$t_2 t_1 :: \texttt{t} \quad \rightarrow \quad \langle t_1, t_2 \rangle :: \langle \texttt{+k t}, \texttt{-k} \rangle$$

$$\textit{Mary will think John will eat cake} :: \texttt{t} \quad \rightarrow \quad \langle \textit{will think John will eat cake}, \textit{Mary} \rangle :: \langle \texttt{+k t}, \texttt{-k} \rangle$$

$$\langle s t_1, t_2 \rangle :: \langle \texttt{+k t}, \texttt{-k} \rangle \quad \rightarrow \quad s :: \texttt{=v +k t} \quad \langle t_1, t_2 \rangle :: \langle \texttt{v}, \texttt{-k} \rangle$$

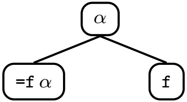
$$\langle \textit{will think John will eat cake}, \textit{Mary} \rangle :: \langle \texttt{+k t}, \texttt{-k} \rangle \rightarrow \textit{will} :: \texttt{=v +k t} \quad \langle \textit{think John will eat cake}, \textit{Mary} \rangle :: \langle \texttt{v}, \texttt{-k} \rangle$$

$$\langle s, t \rangle :: \langle \texttt{v}, \texttt{-k} \rangle \quad \rightarrow \quad s :: \texttt{=subj v} \quad t :: \texttt{subj -k}$$

$$\langle \textit{think John will eat cake}, \textit{Mary} \rangle :: \langle \texttt{v}, \texttt{-k} \rangle \quad \rightarrow \quad \textit{think John will eat cake} :: \texttt{=subj v} \quad \textit{Mary} :: \texttt{subj -k}$$

One slightly annoying wrinkle

We know that this is a valid derivational step:



What is the corresponding MCFG rule?

Selected thing on the right?

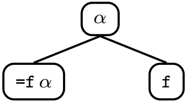
$$st :: \alpha \quad \rightarrow \quad s :: \texttt{=f}\ \alpha \quad t :: \texttt{f}$$

Selected thing on the left?

$$ts :: \alpha \quad \rightarrow \quad s :: \texttt{=f}\ \alpha \quad t :: \texttt{f}$$

One slightly annoying wrinkle

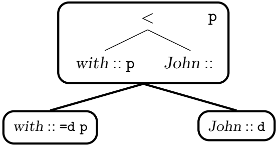
We know that this is a valid derivational step:



What is the corresponding MCFG rule?

Selected thing on the right?

$$st :: \alpha \quad \rightarrow \quad s :: \texttt{=f}\ \alpha \quad t :: \texttt{f}$$

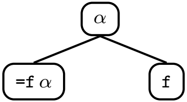


Selected thing on the left?

$$ts :: \alpha \quad \rightarrow \quad s :: \texttt{=f}\ \alpha \quad t :: \texttt{f}$$

One slightly annoying wrinkle

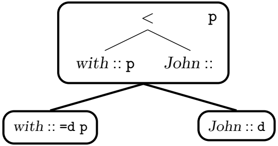
We know that this is a valid derivational step:



What is the corresponding MCFG rule?

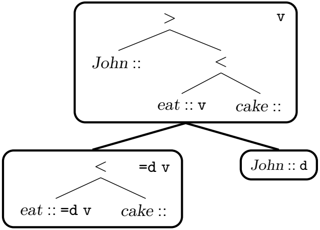
Selected thing on the right?

$st :: \alpha \quad \rightarrow \quad s :: \texttt{=f}\ \alpha \quad t :: \texttt{f}$



Selected thing on the left?

$ts :: \alpha \quad \rightarrow \quad s :: \texttt{=f}\ \alpha \quad t :: \texttt{f}$



One slightly annoying wrinkle

Each type needs to record not only the unchecked features, but also whether the expression is lexical.

I'll write lexical types as $\langle \dots \rangle_1$ and non-lexical types as $\langle \dots \rangle_0$.

So types of the form $\langle \texttt{=f}\ \alpha \rangle_1$ act slightly differently from those of the form $\langle \texttt{=f}\ \alpha \rangle_0$.

$$\begin{array}{rcl} \textit{st} :: \langle \alpha \rangle_0 & \rightarrow & \textit{s} :: \langle \texttt{=f}\ \alpha \rangle_1 \quad \textit{t} :: \langle \texttt{f} \rangle_n \\ \textit{with John} :: \langle \texttt{p} \rangle_0 & \rightarrow & \textit{with} :: \langle \texttt{=d p} \rangle_1 \quad \textit{John} :: \langle \texttt{d} \rangle_1 \end{array}$$

$$\begin{array}{rcl} \textit{ts} :: \langle \alpha \rangle_0 & \rightarrow & \textit{s} :: \langle \texttt{=f}\ \alpha \rangle_0 \quad \textit{t} :: \langle \texttt{f} \rangle_n \\ \textit{John eat cake} :: \langle \texttt{v} \rangle_0 & \rightarrow & \textit{eat cake} :: \langle \texttt{=d v} \rangle_0 \quad \textit{John} :: \langle \texttt{d} \rangle_1 \end{array}$$

Context-free structure

$$\begin{aligned}
\langle \texttt{=subj v} \rangle &\rightarrow \langle \texttt{=q =subj v} \rangle \quad \langle \texttt{q} \rangle \\
\langle \texttt{q} \rangle &\rightarrow \langle \texttt{+wh q}, \texttt{-wh} \rangle \\
\langle \texttt{+wh q}, \texttt{-wh} \rangle &\rightarrow \langle \texttt{=t +wh q} \rangle \quad \langle \texttt{t}, \texttt{-wh} \rangle
\end{aligned}$$

Context-free structure

$$
\begin{array}{rcll}
\langle \texttt{=subj v} \rangle & \rightarrow & \langle \texttt{=q =subj v} \rangle & \langle \texttt{q} \rangle \\
\langle \texttt{q} \rangle & \rightarrow & \langle \texttt{+wh q}, \texttt{-wh} \rangle & \\
\langle \texttt{+wh q}, \texttt{-wh} \rangle & \rightarrow & \langle \texttt{=t +wh q} \rangle & \langle \texttt{t}, \texttt{-wh} \rangle
\end{array}
$$

General schemas for MERGE steps (approximate):

$$
\begin{array}{rcll}
\langle \gamma, \alpha_1, \ldots, \alpha_j, \beta_1, \ldots, \beta_k \rangle & \rightarrow & \langle \texttt{=f}\gamma, \alpha_1, \ldots, \alpha_j \rangle & \langle \texttt{f}, \beta_1, \ldots, \beta_k \rangle \\
\langle \gamma, \alpha_1, \ldots, \alpha_j, \delta, \beta_1, \ldots, \beta_k \rangle & \rightarrow & \langle \texttt{=f}\gamma, \alpha_1, \ldots, \alpha_j \rangle & \langle \texttt{f}\delta, \beta_1, \ldots, \beta_k \rangle
\end{array}
$$

General schemas for MOVE steps (approximate):

$$
\begin{array}{rcl}
\langle \gamma, \alpha_1, \ldots, \alpha_{i-1}, \alpha_{i+1}, \ldots, \alpha_k \rangle & \rightarrow & \langle \texttt{+f}\gamma, \alpha_1, \ldots, \alpha_{i-1}, \texttt{-f}, \alpha_{i+1}, \ldots, \alpha_k \rangle \\
\langle \gamma, \alpha_1, \ldots, \alpha_{i-1}, \delta, \alpha_{i+1}, \ldots, \alpha_k \rangle & \rightarrow & \langle \texttt{+f}\gamma, \alpha_1, \ldots, \alpha_{i-1}, \texttt{-f}\delta, \alpha_{i+1}, \ldots, \alpha_k \rangle
\end{array}
$$

Context-free structure

$$\begin{array}{rcll} \langle \texttt{=subj v} \rangle & \rightarrow & \langle \texttt{=q =subj v} \rangle & \langle \texttt{q} \rangle \\ \langle \texttt{q} \rangle & \rightarrow & \langle \texttt{+wh q}, \texttt{-wh} \rangle & \\ \langle \texttt{+wh q}, \texttt{-wh} \rangle & \rightarrow & \langle \texttt{=t +wh q} \rangle & \langle \texttt{t}, \texttt{-wh} \rangle \end{array}$$

General schemas for MERGE steps (approximate):

$$\begin{array}{rcll} \langle \gamma, \alpha_1, \dots, \alpha_j, \beta_1, \dots, \beta_k \rangle & \rightarrow & \langle \texttt{=f}\gamma, \alpha_1, \dots, \alpha_j \rangle & \langle \texttt{f}, \beta_1, \dots, \beta_k \rangle \\ \langle \gamma, \alpha_1, \dots, \alpha_j, \delta, \beta_1, \dots, \beta_k \rangle & \rightarrow & \langle \texttt{=f}\gamma, \alpha_1, \dots, \alpha_j \rangle & \langle \texttt{f}\delta, \beta_1, \dots, \beta_k \rangle \end{array}$$

General schemas for MOVE steps (approximate):

$$\begin{array}{rcl} \langle \gamma, \alpha_1, \dots, \alpha_{i-1}, \alpha_{i+1}, \dots, \alpha_k \rangle & \rightarrow & \langle \texttt{+f}\gamma, \alpha_1, \dots, \alpha_{i-1}, \texttt{-f}, \alpha_{i+1}, \dots, \alpha_k \rangle \\ \langle \gamma, \alpha_1, \dots, \alpha_{i-1}, \delta, \alpha_{i+1}, \dots, \alpha_k \rangle & \rightarrow & \langle \texttt{+f}\gamma, \alpha_1, \dots, \alpha_{i-1}, \texttt{-f}\delta, \alpha_{i+1}, \dots, \alpha_k \rangle \end{array}$$

- MOVE steps **change** something without **combining** it with anything
- Compare with unary CFG rules, or type-raising in CCG, or ...

Three schemas for MERGE rules:

$$\langle st, t_1, \ldots, t_k \rangle :: \langle \gamma, \alpha_1, \ldots, \alpha_k \rangle_0 \quad \rightarrow \quad s :: \langle \texttt{=f}\gamma \rangle_1 \quad \langle t, t_1, \ldots, t_k \rangle :: \langle \texttt{f}, \alpha_1, \ldots, \alpha_k \rangle_n$$

$$\langle ts, s_1, \ldots, s_j, t_1, \ldots, t_k \rangle :: \langle \gamma, \alpha_1, \ldots, \alpha_j, \beta_1, \ldots, \beta_k \rangle_0 \quad \rightarrow \quad \langle s, s_1, \ldots, s_j \rangle :: \langle \texttt{=f}\gamma, \alpha_1, \ldots, \alpha_j \rangle_0 \quad \langle t, t_1, \ldots, t_k \rangle :: \langle \texttt{f}, \beta_1, \ldots, \beta_k \rangle_n$$

$$\langle s, s_1, \ldots, s_j, t, t_1, \ldots, t_k \rangle :: \langle \gamma, \alpha_1, \ldots, \alpha_j, \delta, \beta_1, \ldots, \beta_k \rangle_0 \quad \rightarrow \quad \langle s, s_1, \ldots, s_j \rangle :: \langle \texttt{=f}\gamma, \alpha_1, \ldots, \alpha_j \rangle_n \quad \langle t, t_1, \ldots, t_k \rangle :: \langle \texttt{f}\delta, \beta_1, \ldots, \beta_k \rangle_{n'}$$

Two schemas for MOVE rules:

$$\langle s_i s, s_1, \ldots, s_{i-1}, s_{i+1}, \ldots, s_k \rangle :: \langle \gamma, \alpha_1, \ldots, \alpha_{i-1}, \alpha_{i+1}, \ldots, \alpha_k \rangle_0 \quad \rightarrow \quad \langle s, s_1, \ldots, s_i, \ldots, s_k \rangle :: \langle \texttt{+f}\gamma, \alpha_1, \ldots, \alpha_{i-1}, \texttt{-f}, \alpha_{i+1}, \ldots, \alpha_k \rangle_0$$

$$\langle s, s_1, \ldots, s_i, \ldots, s_k \rangle :: \langle \gamma, \alpha_1, \ldots, \alpha_{i-1}, \delta, \alpha_{i+1}, \ldots, \alpha_k \rangle_0 \quad \rightarrow \quad \langle s, s_1, \ldots, s_i, \ldots, s_k \rangle :: \langle \texttt{+f}\gamma, \alpha_1, \ldots, \alpha_{i-1}, \texttt{-f}\delta, \alpha_{i+1}, \ldots, \alpha_k \rangle_0$$

Part 4

Probabilities on MG Derivations

Outline

Outline

Probabilistic CFGs

"What are the probabilities of the derivations?"

=

"What are the values of λ_1, λ_2, etc.?"

λ_1	S $\rightarrow$ NP VP
λ_2	NP $\rightarrow$ John
λ_3	NP $\rightarrow$ Mary
λ_4	VP $\rightarrow$ ran
λ_5	VP $\rightarrow$ V NP
λ_6	VP $\rightarrow$ V S
λ_7	V $\rightarrow$ believed
λ_8	V $\rightarrow$ knew

Probabilistic CFGs

"What are the probabilities of the derivations?"

=

"What are the values of λ_1, λ_2, etc.?"

λ_1	S → NP VP
λ_2	NP → John
λ_3	NP → Mary
λ_4	VP → ran
λ_5	VP → V NP
λ_6	VP → V S
λ_7	V → believed
λ_8	V → knew

→ Training algorithm (← Training corpus) →

1.0	S → NP VP
0.3	NP → John
0.7	NP → Mary
0.2	VP → ran
0.5	VP → V NP
0.3	VP → V S
0.4	V → believed
0.6	V → knew

$$\lambda_5 = \frac{\text{count}(\text{VP} \to \text{V NP})}{\text{count}(\text{VP})}$$

MCFG for an entire Minimalist Grammar

Lexical items:

$\epsilon :: \langle \texttt{=t +wh c} \rangle_1$

$\epsilon :: \langle \texttt{=t c} \rangle_1$

$\text{will} :: \langle \texttt{=v =d t} \rangle_1$

$\text{often} :: \langle \texttt{=v v} \rangle_1$

$\text{praise} :: \langle \texttt{=d v} \rangle_1$

$\text{marie} :: \langle \texttt{d} \rangle_1$

$\text{pierre} :: \langle \texttt{d} \rangle_1$

$\text{who} :: \langle \texttt{d -wh} \rangle_1$

Production rules:

$$\begin{aligned}
\langle st, u \rangle :: \langle \texttt{+wh c}, \texttt{-wh} \rangle_0 &\rightarrow s :: \langle \texttt{=t +wh c} \rangle_1 \quad \langle t, u \rangle :: \langle \texttt{t}, \texttt{-wh} \rangle_0 \\
st :: \langle \texttt{=d t} \rangle_0 &\rightarrow s :: \langle \texttt{=v =d t} \rangle_1 \quad t :: \langle \texttt{v} \rangle_0 \\
\langle st, u \rangle :: \langle \texttt{=d t}, \texttt{-wh} \rangle_0 &\rightarrow s :: \langle \texttt{=v =d t} \rangle_1 \quad \langle t, u \rangle :: \langle \texttt{v}, \texttt{-wh} \rangle_0 \\
ts :: \langle \texttt{c} \rangle_0 &\rightarrow \langle s, t \rangle :: \langle \texttt{+wh c}, \texttt{-wh} \rangle_0 \\
st :: \langle \texttt{c} \rangle_0 &\rightarrow s :: \langle \texttt{=t c} \rangle_1 \quad t :: \langle \texttt{t} \rangle_0 \\
ts :: \langle \texttt{t} \rangle_0 &\rightarrow s :: \langle \texttt{=d t} \rangle_0 \quad t :: \langle \texttt{d} \rangle_1 \\
\langle ts, u \rangle :: \langle \texttt{t}, \texttt{-wh} \rangle_0 &\rightarrow \langle s, u \rangle :: \langle \texttt{=d t}, \texttt{-wh} \rangle_0 \quad t :: \langle \texttt{d} \rangle_1 \\
st :: \langle \texttt{v} \rangle_0 &\rightarrow s :: \langle \texttt{=d v} \rangle_1 \quad t :: \langle \texttt{d} \rangle_1 \\
st :: \langle \texttt{v} \rangle_0 &\rightarrow s :: \langle \texttt{=v v} \rangle_1 \quad t :: \langle \texttt{v} \rangle_0 \\
\langle s, t \rangle :: \langle \texttt{v}, \texttt{-wh} \rangle_0 &\rightarrow s :: \langle \texttt{=d v} \rangle_1 \quad t :: \langle \texttt{d -wh} \rangle_1 \\
\langle st, u \rangle :: \langle \texttt{v}, \texttt{-wh} \rangle_0 &\rightarrow s :: \langle \texttt{=v v} \rangle_1 \quad \langle t, u \rangle :: \langle \texttt{v}, \texttt{-wh} \rangle_0
\end{aligned}$$

Probabilities on MCFGs

$$
\begin{array}{llcl}
\lambda_1 & ts :: \langle \texttt{c} \rangle_0 & \to & \langle s, t \rangle :: \langle \texttt{+wh c}, \texttt{-wh} \rangle_0 \\
\lambda_2 & st :: \langle \texttt{c} \rangle_0 & \to & s :: \langle \texttt{=t c} \rangle_1 \quad t :: \langle \texttt{t} \rangle_0 \\
\lambda_3 & st :: \langle \texttt{v} \rangle_0 & \to & s :: \langle \texttt{=d v} \rangle_1 \quad t :: \langle \texttt{d} \rangle_1 \\
\lambda_4 & st :: \langle \texttt{v} \rangle_0 & \to & s :: \langle \texttt{=v v} \rangle_1 \quad t :: \langle \texttt{v} \rangle_0 \\
\lambda_5 & \langle s, t \rangle :: \langle \texttt{v}, \texttt{-wh} \rangle_0 & \to & s :: \langle \texttt{=d v} \rangle_1 \quad t :: \langle \texttt{d -wh} \rangle_1 \\
\lambda_6 & \langle st, u \rangle :: \langle \texttt{v}, \texttt{-wh} \rangle_0 & \to & s :: \langle \texttt{=v v} \rangle_1 \quad \langle t, u \rangle :: \langle \texttt{v}, \texttt{-wh} \rangle_0
\end{array}
$$

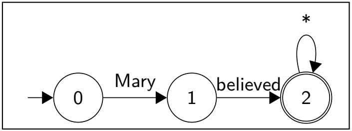
The context-free "backbone" for MG derivations identifies a parametrization for probability distributions over them.

$$
\lambda_2 = \frac{\text{count}\big(\langle \texttt{c} \rangle_0 \to \langle \texttt{=t c} \rangle_1 \langle \texttt{t} \rangle_0\big)}{\text{count}\big(\langle \texttt{c} \rangle_0\big)}
$$

Plus: It turns out that the intersect-with-an-FSA trick we used for CFGs also works for MCFGs!

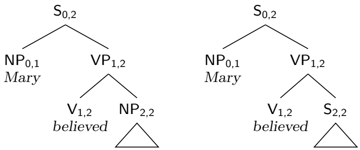
Grammar intersection example (simple)

1.0	S → NP VP
0.3	NP → John
0.7	NP → Mary
0.2	VP → ran
0.5	VP → V NP
0.3	VP → V S
0.4	V → believed
0.6	V → knew



1.0	$S_{0,2}$	→	$NP_{0,1}$ $VP_{1,2}$
0.7	$NP_{0,1}$	→	Mary
0.5	$VP_{1,2}$	→	$V_{1,2}$ $NP_{2,2}$
0.3	$VP_{1,2}$	→	$V_{1,2}$ $S_{2,2}$
0.4	$V_{1,2}$	→	believed

1.0	$S_{2,2}$	→	$NP_{2,2}$ $VP_{2,2}$
0.3	$NP_{2,2}$	→	John
0.7	$NP_{2,2}$	→	Mary
0.2	$VP_{2,2}$	→	ran
0.5	$VP_{2,2}$	→	$V_{2,2}$ $NP_{2,2}$
0.3	$VP_{2,2}$	→	$V_{2,2}$ $S_{2,2}$
0.4	$V_{2,2}$	→	believed
0.6	$V_{2,2}$	→	knew

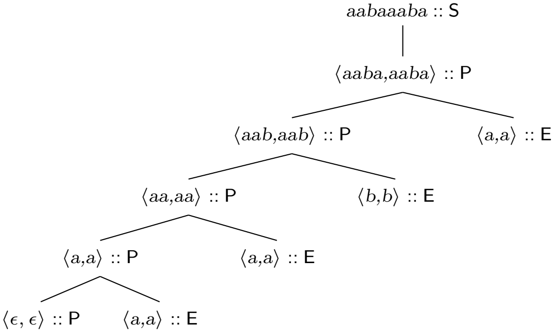


NB: Total weight in this grammar is not one! (What is it? Start symbol is $S_{0,2}$.)
Each derivation has the weight "it" had in the original grammar.

Beyond context-free

$$
\begin{aligned}
t_1 t_2 :: \mathsf{S} &\rightarrow \langle t_1, t_2 \rangle :: \mathsf{P} \\
\langle t_1 u_1, t_2 u_2 \rangle :: \mathsf{P} &\rightarrow \langle t_1, t_2 \rangle :: \mathsf{P} \quad \langle u_1, u_2 \rangle :: \mathsf{E} \\
\langle \epsilon, \epsilon \rangle &:: \mathsf{P} \\
\langle a, a \rangle &:: \mathsf{E} \\
\langle b, b \rangle &:: \mathsf{E}
\end{aligned}
$$

$$\{ww \mid w \in \{a, b\}^*\}$$



Unlike in a CFG, we can ensure that the two "halves" are extended in the same ways without concatenating them together.

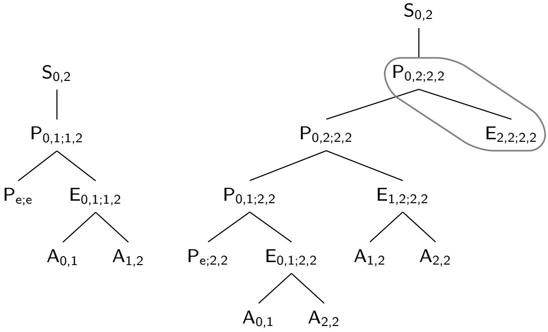
Intersection with an MCFG

$$\begin{array}{lcl} S_{0,2} & \to & P_{0,1;1,2} \\ P_{0,1;1,2} & \to & P_{e;e}\ E_{0,1;1,2} \\ E_{0,1;1,2} & \to & A_{0,1}\ A_{1,2} \end{array}$$

$$\begin{array}{lcl} S_{0,2} & \to & P_{0,2;2,2} \\ P_{0,2;2,2} & \to & P_{0,2;2,2}\ E_{2,2;2,2} \\ P_{0,2;2,2} & \to & P_{0,1;2,2}\ E_{1,2;2,2} \\ P_{0,1;2,2} & \to & P_{e;2,2}\ E_{0,1;2,2} \\ E_{0,1;2,2} & \to & A_{0,1}\ A_{2,2} \\ E_{1,2;2,2} & \to & A_{1,2}\ A_{2,2} \end{array}$$

$\langle b,b\rangle :: E_{2,2;2,2}$
$\langle a,a\rangle :: E_{2,2;2,2}$
$\langle \epsilon, \epsilon\rangle :: P_{e;e}$
$\langle \epsilon, \epsilon\rangle :: P_{e;2,2}$
$a :: A_{2,2}$
$b :: B_{2,2}$

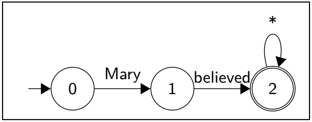
$a :: A_{0,1}$
$a :: A_{1,2}$



Intersection grammars

1.0	S → NP VP
0.3	NP → John
0.7	NP → Mary
0.2	VP → ran
0.5	VP → V NP
0.3	VP → V S
0.4	V → believed
0.6	V → knew

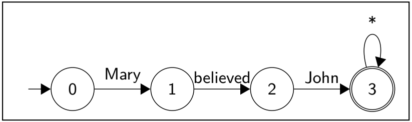
$\cap$



$= \quad G_2$

1.0	S → NP VP
0.3	NP → John
0.7	NP → Mary
0.2	VP → ran
0.5	VP → V NP
0.3	VP → V S
0.4	V → believed
0.6	V → knew

$\cap$



$= \quad G_3$

$$\begin{aligned}
\text{surprisal at `John'} &= -\log P(W_3 = \text{John} \mid W_1 = \text{Mary}, W_2 = \text{believed}) \\
&= -\log \frac{\text{total weight in } G_3}{\text{total weight in } G_2} \\
&= -\log \frac{0.0672}{0.224} \\
&= 1.74
\end{aligned}$$

Surprisal and entropy reduction

$$\text{surprisal at ‘John’} = -\log P(W_3 = \text{John} \mid W_1 = \text{Mary}, W_2 = \text{believed})$$
$$= -\log \frac{\text{total weight in } G_3}{\text{total weight in } G_2}$$

$$\text{entropy reduction at ‘John’} = (\text{entropy of } G_2) - (\text{entropy of } G_3)$$

Computing sum of weights in a grammar ("partition function")

$$Z(A) = \sum_{A \to \alpha} \Big(p(A \to \alpha) \cdot Z(\alpha) \Big)$$
$$Z(\epsilon) = 1$$
$$Z(a\beta) = Z(\beta)$$
$$Z(B\beta) = Z(B) \cdot Z(\beta) \qquad \text{where } \beta \neq \epsilon$$

(Nederhof and Satta 2008)

1.0	S $\to$ NP VP
0.3	NP $\to$ John
0.7	NP $\to$ Mary
0.2	VP $\to$ ran
0.5	VP $\to$ V NP
0.4	V $\to$ believed
0.6	V $\to$ knew

$$\begin{aligned} Z(\text{V}) &= 0.4 + 0.6 = 1.0 \\ Z(\text{NP}) &= 0.3 + 0.7 = 1.0 \\ Z(\text{VP}) &= 0.2 + (0.5 \cdot Z(\text{V}) \cdot Z(\text{NP})) \\ &= 0.2 + (0.5 \cdot 1.0 \cdot 1.0) = 0.7 \\ Z(\text{S}) &= 1.0 \cdot Z(\text{NP}) \cdot Z(\text{VP}) \\ &= 0.7 \end{aligned}$$

1.0	S $\to$ NP VP
0.3	NP $\to$ John
0.7	NP $\to$ Mary
0.2	VP $\to$ ran
0.5	VP $\to$ V NP
0.3	VP $\to$ V S
0.4	V $\to$ believed
0.6	V $\to$ knew

$$\begin{aligned} Z(\text{V}) &= 0.4 + 0.6 = 1.0 \\ Z(\text{NP}) &= 0.3 + 0.7 = 1.0 \\ Z(\text{VP}) &= 0.2 + (0.5 \cdot Z(\text{V}) \cdot Z(\text{NP})) + (0.3 \cdot Z(\text{V}) \cdot Z(\text{S})) \\ Z(\text{S}) &= 1.0 \cdot Z(\text{NP}) \cdot Z(\text{VP}) \end{aligned}$$

Computing entropy of a grammar

1.0	S → NP VP
0.3	NP → John
0.7	NP → Mary
0.2	VP → ran
0.5	VP → V NP
0.3	VP → V S
0.4	V → believed
0.6	V → knew

$$h(\text{S}) = 0$$
$$h(\text{NP}) = \text{entropy of } (0.3, 0.7)$$
$$h(\text{VP}) = \text{entropy of } (0.2, 0.5, 0.3)$$
$$h(\text{V}) = \text{entropy of } (0.4, 0.6)$$

$$H(\text{S}) = h(\text{S}) + 1.0(H(\text{NP}) + H(\text{VP}))$$
$$H(\text{NP}) = h(\text{NP})$$
$$H(\text{VP}) = h(\text{VP}) + 0.2(0) + 0.5(H(\text{V}) + H(\text{NP})) + 0.3(H(\text{V}) + H(\text{S}))$$
$$H(\text{V}) = h(\text{V})$$

(Hale 2006)

Surprisal and entropy reduction

$$\begin{aligned}\text{surprisal at `John'} &= -\log P(W_3 = \text{John} \mid W_1 = \text{Mary}, W_2 = \text{believed}) \\ &= -\log \frac{\text{total weight in } G_3}{\text{total weight in } G_2}\end{aligned}$$

$$\text{entropy reduction at `John'} = (\text{entropy of } G_2) - (\text{entropy of } G_3)$$

Putting it all together (Hale 2006)

We can now put entropy reduction/surprisal together with a minimalist grammar to produce predictions about sentence comprehension difficulty!

complexity metric $+$ grammar $\longrightarrow$ prediction

- Write an MG that generates sentence types of interest
- Convert MG to an MCFG
- Add probabilities to MCFG based on corpus frequencies (or whatever else)
- Compute intersection grammars for each point in a sentence
- Calculate reduction in entropy across the course of the sentence (i.e. workload)

Demo

Hale (2006)

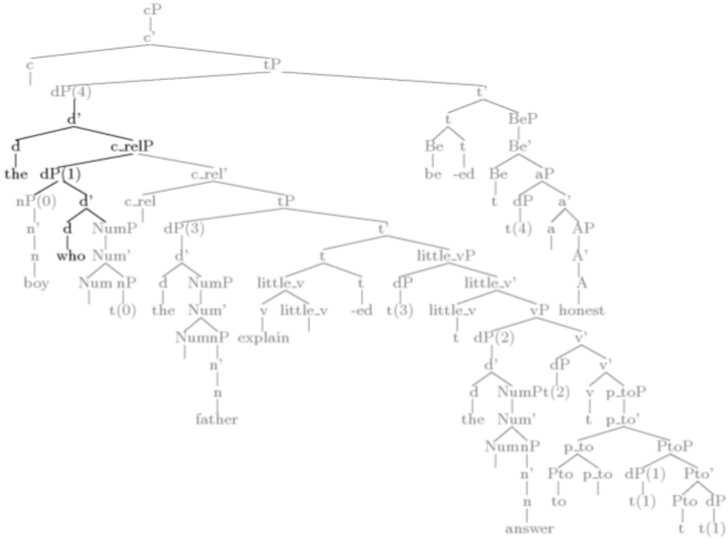


Fig. 11. Kaynian promotion analysis.

Hale (2006)

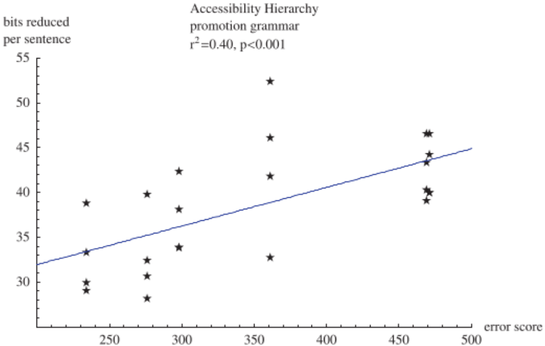
```
they have -ed forget -en that the boy who tell -ed the story be -s so young
the fact that the girl who pay -ed for the ticket be -s very poor doesnt matter
I know that the girl who get -ed the right answer be -s clever
he remember -ed that the man who sell -ed the house leave -ed the town

they have -ed forget -en that the letter which Dick write -ed yesterday be -s long
the fact that the cat which David show -ed to the man like -s eggs be -s strange
I know that the dog which Penny buy -ed today be -s very gentle
he remember -ed that the sweet which David give -ed Sally be -ed a treat

they have -ed forget -en that the man who Ann give -ed the present to be -ed old
the fact that the boy who Paul sell -ed the book to hate -s reading be -s strange
I know that the man who Stephen explain -ed the accident to be -s kind
he remember -ed that the dog which Mary teach -ed the trick to be -s clever

they have -ed forget -en that the box which Pat bring -ed the apple in be -ed lost
the fact that the girl who Sue write -ed the story with be -s proud doesnt matter
I know that the ship which my uncle take -ed Joe on be -ed interesting
he remember -ed that the food which Chris pay -ed the bill for be -ed cheap

they have -ed forget -en that the girl whose friend buy -ed the cake be -ed wait -ing
the fact that the boy whose brother tell -s lies be -s always honest surprise -ed us
I know that the boy whose father sell -ed the dog be -ed very sad
he remember -ed that the girl whose mother send -ed the clothe come -ed too late

they have -ed forget -en that the man whose house Patrick buy -ed be -ed so ill
the fact that the sailor whose ship Jim take -ed have -ed one leg be -s important
I know that the woman whose car Jenny sell -ed be -ed very angry
he remember -ed that the girl whose picture Clare show -ed us be -ed pretty
```

Hale (2006)

count	grammatical relation	definition
1430	subject	co-indexed trace is the first daughter of S
929	direct object	co-indexed trace is immediately following sister of a V-node
167	indirect object	co-indexed trace is part of a PP not annotated as benefactive, locative, manner, purpose, temporal or directional
41	oblique	co-indexed trace is part of a benefactive, locative, manner, purpose, temporal or directional PP
34	genitive subject	WH word is *whose* and co-indexed trace is first daughter of S
4	genitive direct object	WH word is *whose* and co-indexed trace is immediately following sister of a V-node

Fig. 13. Counts from Brown portion of Penn Treebank III.

Hale (2006)



(a) Per-sentence entropy reductions on the promotion grammar

Grammatical Relation:	SU	DO	IO	OBL	GenS	GenO
Repetition Accuracy:	406	364	342	279	167	171
errors (= $R.A._{max} - R.A.$)	234	276	298	361	471	469

Fig. 8. Results from Keenan and Hawkins (1987).

Hale (2006)

Hale actually wrote two different MGs:

- classical adjunction analysis of relative clauses
- Kaynian/promotion analysis

Hale (2006)

Hale actually wrote two different MGs:

- classical adjunction analysis of relative clauses
- Kaynian/promotion analysis

The branching structure of the two MCFGs was different enough to produce distinct Entropy Reduction predictions. (Same corpus counts!)

The Kaynian/promotion analysis produced a better fit for the Accessibility Hierarchy facts.
(i.e. holding the complexity metric fixed to argue for a grammar)

But there are some ways in which this method is insensitive to fine details of the MG formalism.

Outline

Subtlely different minimalist frameworks

Minimalist grammars with many choices of different bells and whistles can all be expressed with context-free derivational structure.

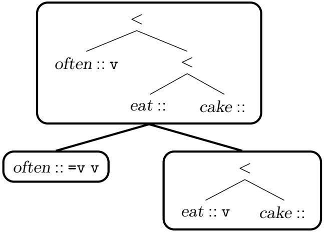
- Must keep an eye on finiteness of number of types (SMC or equivalent)!
- See Stabler (2011)

Some points of variation:

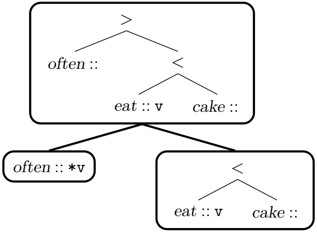
- adjunction
- head movement
- phases
- move as re-merge
- ...

How to deal with adjuncts?

A normal application of MERGE?

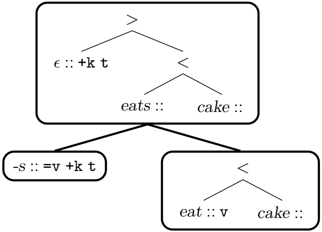


Or a new kind of feature and distinct operation ADJOIN?



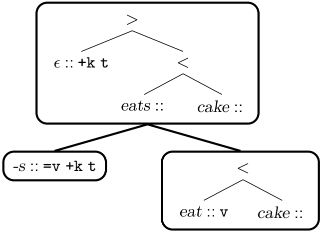
How to implement "head movement"?

Modify MERGE to allow some additional string-shuffling in head-complement relationships?

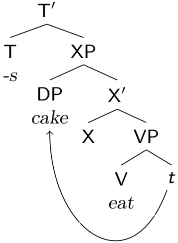


How to implement "head movement"?

Modify MERGE to allow some additional string-shuffling in head-complement relationships?

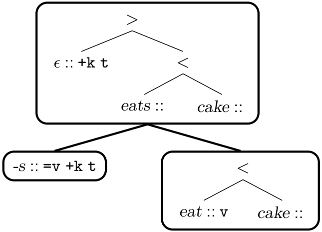


Or some combination of normal phrasal movements? (Koopman and Szabolcsi 2000)

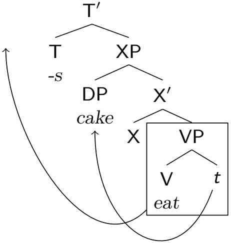


How to implement "head movement"?

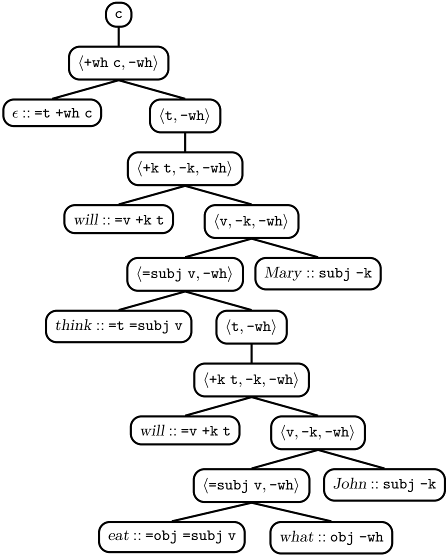
Modify MERGE to allow some additional string-shuffling in head-complement relationships?



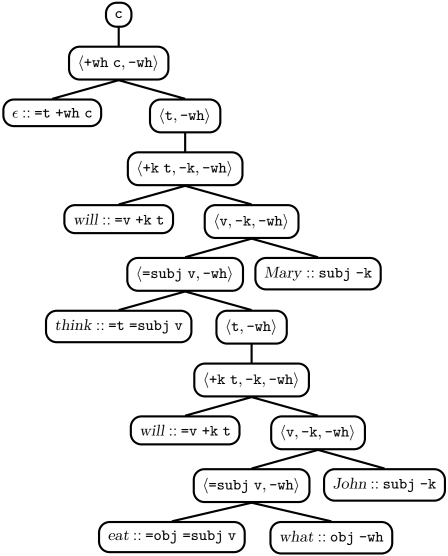
Or some combination of normal phrasal movements? (Koopman and Szabolcsi 2000)

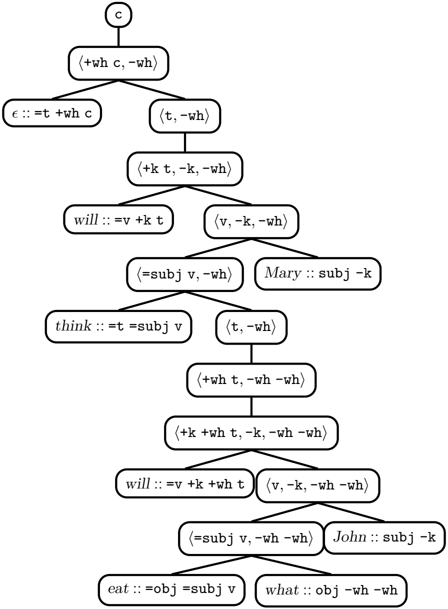


Successive cyclic movement?

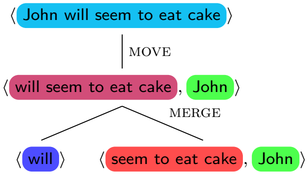


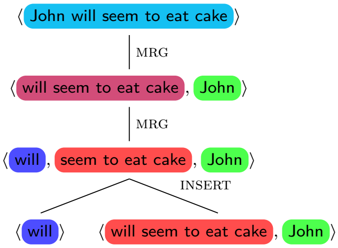
Successive cyclic movement?





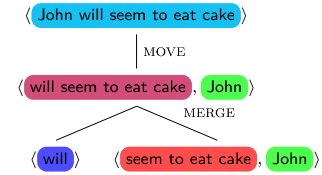
Unifying feature-checking (one way)

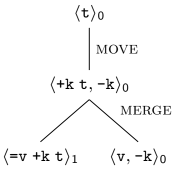


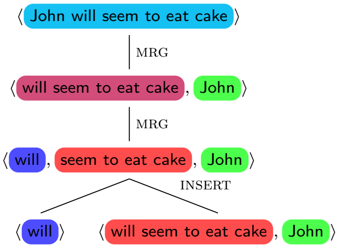


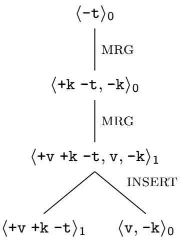
(Stabler 2006, Hunter 2011)

Unifying feature-checking (one way)









(Stabler 2006, Hunter 2011)

Three schemas for MERGE rules:

$$\langle st, t_1, \ldots, t_k \rangle :: \langle \gamma, \alpha_1, \ldots, \alpha_k \rangle_0 \quad \rightarrow \quad s :: \langle \texttt{=f}\gamma \rangle_1 \quad \langle t, t_1, \ldots, t_k \rangle :: \langle \texttt{f}, \alpha_1, \ldots, \alpha_k \rangle_n$$

$$\langle ts, s_1, \ldots, s_j, t_1, \ldots, t_k \rangle :: \langle \gamma, \alpha_1, \ldots, \alpha_j, \beta_1, \ldots, \beta_k \rangle_0 \quad \rightarrow \quad \langle s, s_1, \ldots, s_j \rangle :: \langle \texttt{=f}\gamma, \alpha_1, \ldots, \alpha_j \rangle_0 \quad \langle t, t_1, \ldots, t_k \rangle :: \langle \texttt{f}, \beta_1, \ldots, \beta_k \rangle_n$$

$$\langle s, s_1, \ldots, s_j, t, t_1, \ldots, t_k \rangle :: \langle \gamma, \alpha_1, \ldots, \alpha_j, \delta, \beta_1, \ldots, \beta_k \rangle_0 \quad \rightarrow \quad \langle s, s_1, \ldots, s_j \rangle :: \langle \texttt{=f}\gamma, \alpha_1, \ldots, \alpha_j \rangle_n \quad \langle t, t_1, \ldots, t_k \rangle :: \langle \texttt{f}\delta, \beta_1, \ldots, \beta_k \rangle_{n'}$$

Two schemas for MOVE rules:

$$\langle s_i s, s_1, \ldots, s_{i-1}, s_{i+1}, \ldots, s_k \rangle :: \langle \gamma, \alpha_1, \ldots, \alpha_{i-1}, \alpha_{i+1}, \ldots, \alpha_k \rangle_0 \quad \rightarrow \quad \langle s, s_1, \ldots, s_i, \ldots, s_k \rangle :: \langle \texttt{+f}\gamma, \alpha_1, \ldots, \alpha_{i-1}, \texttt{-f}, \alpha_{i+1}, \ldots, \alpha_k \rangle_0$$

$$\langle s, s_1, \ldots, s_i, \ldots, s_k \rangle :: \langle \gamma, \alpha_1, \ldots, \alpha_{i-1}, \delta, \alpha_{i+1}, \ldots, \alpha_k \rangle_0 \quad \rightarrow \quad \langle s, s_1, \ldots, s_i, \ldots, s_k \rangle :: \langle \texttt{+f}\gamma, \alpha_1, \ldots, \alpha_{i-1}, \texttt{-f}\delta, \alpha_{i+1}, \ldots, \alpha_k \rangle_0$$

One schema for INSERT rules:

$$\langle s, s_1, \ldots, s_j, t, t_1, \ldots, t_k \rangle :: \langle \texttt{+f}\gamma, \alpha_1, \ldots, \alpha_j, \texttt{-f}\gamma', \beta_1, \ldots, \beta_k \rangle_n \quad \rightarrow \quad s, s_1, \ldots, s_j :: \langle \texttt{+f}\gamma, \alpha_1, \ldots, \alpha_j \rangle_n \quad \langle t, t_1, \ldots, t_k \rangle :: \langle \texttt{-f}\gamma', \beta_1, \ldots, \beta_k \rangle_{n'}$$

Three schemas for MRG rules:

$$\langle ss_i, s_1, \ldots, s_{i-1}, s_{i+1}, \ldots, s_k \rangle :: \langle \gamma, \alpha_1, \ldots, \alpha_{i-1}, \alpha_{i+1}, \ldots, \alpha_k \rangle_0 \quad \rightarrow \quad \langle s, s_1, \ldots, s_i, \ldots, s_k \rangle :: \langle \texttt{+f}\gamma, \alpha_1, \ldots, \alpha_{i-1}, \texttt{-f}, \alpha_{i+1}, \ldots, \alpha_k \rangle_1$$

$$\langle s_i s, s_1, \ldots, s_{i-1}, s_{i+1}, \ldots, s_k \rangle :: \langle \gamma, \alpha_1, \ldots, \alpha_{i-1}, \alpha_{i+1}, \ldots, \alpha_k \rangle_0 \quad \rightarrow \quad \langle s, s_1, \ldots, s_i, \ldots, s_k \rangle :: \langle \texttt{+f}\gamma, \alpha_1, \ldots, \alpha_{i-1}, \texttt{-f}, \alpha_{i+1}, \ldots, \alpha_k \rangle_0$$

$$\langle s, s_1, \ldots, s_i, \ldots, s_k \rangle :: \langle \gamma, \alpha_1, \ldots, \alpha_{i-1}, \delta, \alpha_{i+1}, \ldots, \alpha_k \rangle_0 \quad \rightarrow \quad \langle s, s_1, \ldots, s_i, \ldots, s_k \rangle :: \langle \texttt{+f}\gamma, \alpha_1, \ldots, \alpha_{i-1}, \texttt{-f}\delta, \alpha_{i+1}, \ldots, \alpha_k \rangle_0$$

Subtlely different minimalist frameworks

Minimalist grammars with many choices of different bells and whistles can all be expressed with context-free derivational structure.

- Must keep an eye on finiteness of number of types (SMC or equivalent)!
- See Stabler (2011)

Some points of variation:

- adjunction
- head movement
- phases
- move as re-merge
- ...

Subtlely different minimalist frameworks

Minimalist grammars with many choices of different bells and whistles can all be expressed with context-free derivational structure.

- Must keep an eye on finiteness of number of types (SMC or equivalent)!
- See Stabler (2011)

Some points of variation:

- adjunction
- head movement
- phases
- move as re-merge
- ...

Each variant of the formalism expresses a different hypothesis about the set of primitive grammatical operations. (We are looking for ways to tell these apart!)

- The "shapes" of the derivation trees are generally very similar from one variant to the next.
- But variants will make different classifications of the derivational steps involved, according to which operation is being applied.

Outline

Probabilities on MCFGs

λ_1	$ts :: \langle \texttt{c} \rangle_0$	$\rightarrow$	$\langle s, t \rangle :: \langle \texttt{+wh c}, \texttt{-wh} \rangle_0$	
λ_2	$st :: \langle \texttt{c} \rangle_0$	$\rightarrow$	$s :: \langle \texttt{=t c} \rangle_1$	$t :: \langle \texttt{t} \rangle_0$
λ_3	$st :: \langle \texttt{v} \rangle_0$	$\rightarrow$	$s :: \langle \texttt{=d v} \rangle_1$	$t :: \langle \texttt{d} \rangle_1$
λ_4	$st :: \langle \texttt{v} \rangle_0$	$\rightarrow$	$s :: \langle \texttt{=v v} \rangle_1$	$t :: \langle \texttt{v} \rangle_0$
λ_5	$\langle s, t \rangle :: \langle \texttt{v}, \texttt{-wh} \rangle_0$	$\rightarrow$	$s :: \langle \texttt{=d v} \rangle_1$	$t :: \langle \texttt{d -wh} \rangle_1$
λ_6	$\langle st, u \rangle :: \langle \texttt{v}, \texttt{-wh} \rangle_0$	$\rightarrow$	$s :: \langle \texttt{=v v} \rangle_1$	$\langle t, u \rangle :: \langle \texttt{v}, \texttt{-wh} \rangle_0$

Training question: What values of λ_1, λ_2, etc. make the training corpus most likely?

Problem #1 with the naive parametrization

The 'often' Grammar: $\mathrm{MG}_{\mathrm{often}}$

pierre :: `d`	*who* :: `d -wh`
marie :: `d`	*will* :: `=v =d t`
praise :: `=d v`	ϵ :: `=t c`
often :: `=v v`	ϵ :: `=t +wh c`

Training data

90	pierre will praise marie
5	pierre will **often** praise marie
1	who pierre will praise
1	who pierre will **often** praise

Problem #1 with the naive parametrization

The 'often' Grammar: MG_{often}

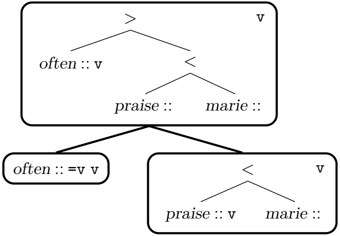
pierre :: d	*who* :: d -wh
marie :: d	*will* :: =v =d t
praise :: =d v	ϵ :: =t c
often :: =v v	ϵ :: =t +wh c

Training data

90	pierre will praise marie
5	pierre will **often** praise marie
1	who pierre will praise
1	who pierre will **often** praise

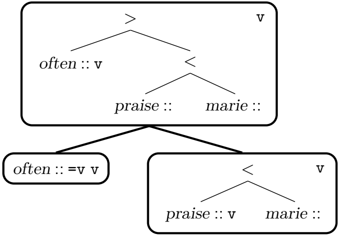
$st :: \langle \mathtt{v} \rangle_0$	$\rightarrow$	$s :: \langle \mathtt{=d\ v} \rangle_1$	$t :: \langle \mathtt{d} \rangle_1$		0.95
$st :: \langle \mathtt{v} \rangle_0$	$\rightarrow$	$s :: \langle \mathtt{=v\ v} \rangle_1$	$t :: \langle \mathtt{v} \rangle_0$		0.05
$\langle s, t \rangle :: \langle \mathtt{v}, \mathtt{-wh} \rangle_0$	$\rightarrow$	$s :: \langle \mathtt{=d\ v} \rangle_1$	$t :: \langle \mathtt{d\ -wh} \rangle_1$		0.67
$\langle st, u \rangle :: \langle \mathtt{v}, \mathtt{-wh} \rangle_0$	$\rightarrow$	$s :: \langle \mathtt{=v\ v} \rangle_1$	$\langle t, u \rangle :: \langle \mathtt{v}, \mathtt{-wh} \rangle_0$		0.33

Generalizations missed by the naive parametrization

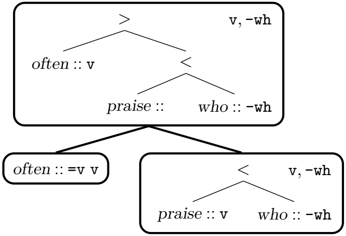


$$\textit{st} :: \langle \texttt{v} \rangle_0 \quad \rightarrow \quad \textit{s} :: \langle \texttt{=v v} \rangle_1 \quad \textit{t} :: \langle \texttt{v} \rangle_0$$

Generalizations missed by the naive parametrization



$$st :: \langle \texttt{v} \rangle_0 \quad \rightarrow \quad s :: \langle \texttt{=v v} \rangle_1 \quad t :: \langle \texttt{v} \rangle_0$$



$$\langle st, u \rangle :: \langle \texttt{v}, \texttt{-wh} \rangle_0 \quad \rightarrow \quad s :: \langle \texttt{=v v} \rangle_1 \quad \langle t, u \rangle :: \langle \texttt{v}, \texttt{-wh} \rangle_0$$

Problem #1 with the naive parametrization

The 'often' Grammar: MG_{often}

pierre :: d	*who* :: d -wh
marie :: d	*will* :: =v =d t
praise :: =d v	ϵ :: =t c
often :: =v v	ϵ :: =t +wh c

Training data

90	pierre will praise marie
5	pierre will **often** praise marie
1	who pierre will praise
1	who pierre will **often** praise

$$
\begin{array}{rcllr}
st :: \langle \mathtt{v} \rangle_0 & \rightarrow & s :: \langle \mathtt{=d\ v} \rangle_1 & t :: \langle \mathtt{d} \rangle_1 & 0.95 \\
st :: \langle \mathtt{v} \rangle_0 & \rightarrow & s :: \langle \mathtt{=v\ v} \rangle_1 & t :: \langle \mathtt{v} \rangle_0 & 0.05 \\
\langle s, t \rangle :: \langle \mathtt{v}, \mathtt{-wh} \rangle_0 & \rightarrow & s :: \langle \mathtt{=d\ v} \rangle_1 & t :: \langle \mathtt{d\ -wh} \rangle_1 & 0.67 \\
\langle st, u \rangle :: \langle \mathtt{v}, \mathtt{-wh} \rangle_0 & \rightarrow & s :: \langle \mathtt{=v\ v} \rangle_1 & \langle t, u \rangle :: \langle \mathtt{v}, \mathtt{-wh} \rangle_0 & 0.33
\end{array}
$$

Problem #1 with the naive parametrization

The 'often' Grammar: MG_{often}

pierre :: d	*who* :: d -wh
marie :: d	*will* :: =v =d t
praise :: =d v	ϵ :: =t c
often :: =v v	ϵ :: =t +wh c

Training data

90	pierre will praise marie
5	pierre will **often** praise marie
1	who pierre will praise
1	who pierre will **often** praise

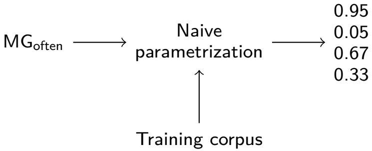
$$
\begin{array}{rcllr}
st :: \langle \texttt{v} \rangle_0 & \rightarrow & s :: \langle \texttt{=d v} \rangle_1 & t :: \langle \texttt{d} \rangle_1 & 0.95 \\
st :: \langle \texttt{v} \rangle_0 & \rightarrow & s :: \langle \texttt{=v v} \rangle_1 & t :: \langle \texttt{v} \rangle_0 & 0.05 \\
\langle s, t \rangle :: \langle \texttt{v}, \texttt{-wh} \rangle_0 & \rightarrow & s :: \langle \texttt{=d v} \rangle_1 & t :: \langle \texttt{d -wh} \rangle_1 & 0.67 \\
\langle st, u \rangle :: \langle \texttt{v}, \texttt{-wh} \rangle_0 & \rightarrow & s :: \langle \texttt{=v v} \rangle_1 & \langle t, u \rangle :: \langle \texttt{v}, \texttt{-wh} \rangle_0 & 0.33
\end{array}
$$

$$
\frac{\text{count}\left(\langle \texttt{v} \rangle_0 \rightarrow \langle \texttt{=d v} \rangle_1 \ \langle \texttt{d} \rangle_1\right)}{\text{count}\left(\langle \texttt{v} \rangle_0\right)} = \frac{95}{100}
$$

$$
\frac{\text{count}\left(\langle \texttt{v,-wh} \rangle_0 \rightarrow \langle \texttt{=d v} \rangle_1 \ \langle \texttt{d -wh} \rangle_1\right)}{\text{count}\left(\langle \texttt{v,-wh} \rangle_0\right)} = \frac{2}{3}
$$

This training setup doesn't know which minimalist-grammar operations are being implemented by the various MCFG rules.

Naive parametrization



Outline

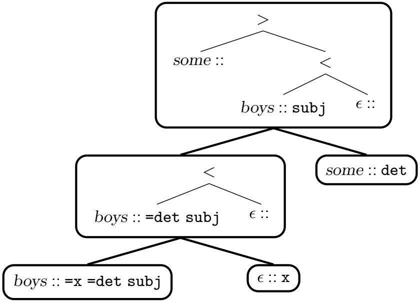
A (slightly) more complicated grammar: MG_{shave}

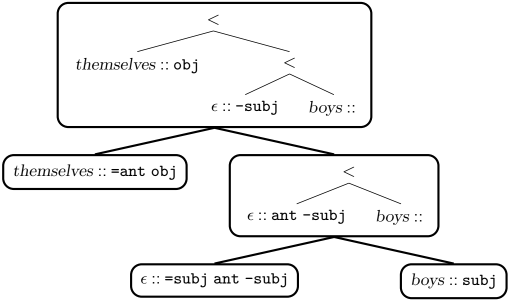
ϵ :: `=t c`	*boys* :: `=x =det subj`
ϵ :: `=t +wh c`	ϵ :: `x`
will :: `=v =subj t`	*some* :: `det`
shave :: `v`	
shave :: `=obj v`	*themselves* :: `=ant obj`
boys :: `subj`	ϵ :: `=subj ant -subj`
who :: `subj -wh`	*will* :: `=v +subj t`

boys will shave
boys will shave themselves
who will shave
who will shave themselves
some boys will shave
some boys will shave themselves

Some details:

- Subject is base-generated in SpecTP; no movement for Case
- Transitive and intransitive versions of *shave*
- *some* is a determiner that optionally combines with *boys* to make a subject
  - Dummy feature `x` to fill complement of *boys* so that *some* goes on the left
- *themselves* can appear in object position, via a movement theory of reflexives
  - A `subj` can be turned into an `ant -subj`
  - *themselves* combines with an `ant` to make an `obj`
  - *will* can attract its subject by move as well as merge


>
some ::
<
boys :: subj
ϵ ::
<
boys :: =det subj
ϵ ::
some :: det
boys :: =x =det subj
ϵ :: x


<
themselves :: obj
<
ϵ :: -subj
boys ::
themselves :: =ant obj
<
ϵ :: ant -subj
boys ::
ϵ :: =subj ant -subj
boys :: subj

Choice points in the MG-derived MCFG

Question or not?			
$\langle \texttt{c} \rangle_0$	$\rightarrow$	$\langle \texttt{=t c} \rangle_0$	$\langle \texttt{t} \rangle_0$
$\langle \texttt{c} \rangle_0$	$\rightarrow$	$\langle \texttt{+wh c}, \texttt{-wh} \rangle_0$	

Antecedent lexical or complex?			
$\langle \texttt{ant -subj} \rangle_0$	$\rightarrow$	$\langle \texttt{=subj ant -subj} \rangle_1$	$\langle \texttt{subj} \rangle_0$
$\langle \texttt{ant -subj} \rangle_0$	$\rightarrow$	$\langle \texttt{=subj ant -subj} \rangle_1$	$\langle \texttt{subj} \rangle_1$

Non-wh subject merged and complex, merged and lexical, or moved?			
$\langle \texttt{t} \rangle_0$	$\rightarrow$	$\langle \texttt{=subj t} \rangle_0$	$\langle \texttt{subj} \rangle_0$
$\langle \texttt{t} \rangle_0$	$\rightarrow$	$\langle \texttt{=subj t} \rangle_0$	$\langle \texttt{subj} \rangle_1$
$\langle \texttt{t} \rangle_0$	$\rightarrow$	$\langle \texttt{+subj t}, \texttt{-subj} \rangle_0$	

Wh-phrase same as moving subject or separated because of doubling?			
$\langle \texttt{t}, \texttt{-wh} \rangle_0$	$\rightarrow$	$\langle \texttt{=subj t} \rangle_0$	$\langle \texttt{subj -wh} \rangle_1$
$\langle \texttt{t}, \texttt{-wh} \rangle_0$	$\rightarrow$	$\langle \texttt{+subj t}, \texttt{-subj}, \texttt{-wh} \rangle_0$	

Choice points in the IMG-derived MCFG

Question or not?

$\langle \texttt{-c} \rangle_0 \rightarrow \langle \texttt{+t -c}, \texttt{-t} \rangle_1$

$\langle \texttt{-c} \rangle_0 \rightarrow \langle \texttt{+wh -c}, \texttt{-wh} \rangle_0$

Antecedent lexical or complex?

$\langle \texttt{+subj -ant -subj}, \texttt{-subj} \rangle_0 \rightarrow \langle \texttt{+subj -ant -subj} \rangle_0 \quad \langle \texttt{-subj} \rangle_0$

$\langle \texttt{+subj -ant -subj}, \texttt{-subj} \rangle_0 \rightarrow \langle \texttt{+subj -ant -subj} \rangle_0 \quad \langle \texttt{-subj} \rangle_1$

Non-wh subject merged and complex, merged and lexical, or moved?

$\langle \texttt{+subj -t}, \texttt{-subj} \rangle_0 \rightarrow \langle \texttt{+subj -t} \rangle_0 \quad \langle \texttt{-subj} \rangle_0$

$\langle \texttt{+subj -t}, \texttt{-subj} \rangle_0 \rightarrow \langle \texttt{+subj -t} \rangle_0 \quad \langle \texttt{-subj} \rangle_1$

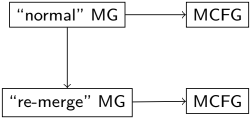
$\langle \texttt{+subj -t}, \texttt{-subj} \rangle_0 \rightarrow \langle \texttt{+v +subj -t}, \texttt{-v}, \texttt{-subj} \rangle_1$

Wh-phrase same as moving subject or separated because of doubling?

$\langle \texttt{-t}, \texttt{-wh} \rangle_0 \rightarrow \langle \texttt{+subj -t}, \texttt{-subj -wh} \rangle_0$

$\langle \texttt{-t}, \texttt{-wh} \rangle_0 \rightarrow \langle \texttt{+subj -t}, \texttt{-subj}, \texttt{-wh} \rangle_0$

Problem #2 with the naive parametrization



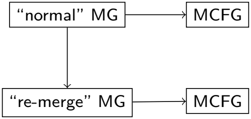
Language of both grammars

boys will shave
boys will shave themselves
who will shave
who will shave themselves
some boys will shave
some boys will shave themselves

Training data

10	boys will shave
2	boys will shave themselves
3	who will shave
1	who will shave themselves
5	some boys will shave

Problem #2 with the naive parametrization



Language of both grammars

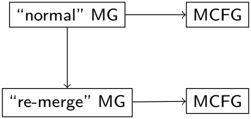
boys will shave
boys will shave themselves
who will shave
who will shave themselves
some boys will shave
some boys will shave themselves

Training data

10	boys will shave
2	boys will shave themselves
3	who will shave
1	who will shave themselves
5	some boys will shave

MG_{shave}, i.e. merge and move distinct	
0.47619	boys will shave
0.238095	some boys will shave
0.142857	who will shave
0.0952381	boys will shave themselves
0.047619	who will shave themselves

Problem #2 with the naive parametrization



Language of both grammars

boys will shave
boys will shave themselves
who will shave
who will shave themselves
some boys will shave
some boys will shave themselves

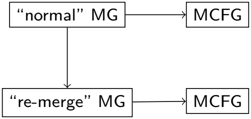
Training data

10	boys will shave
2	boys will shave themselves
3	who will shave
1	who will shave themselves
5	some boys will shave

MG_{shave}, i.e. merge and move distinct	
0.47619	boys will shave
0.238095	some boys will shave
0.142857	who will shave
0.0952381	boys will shave themselves
0.047619	who will shave themselves

IMG_{shave}, i.e. merge and move unified	
0.47619	boys will shave
0.238095	some boys will shave
0.142857	who will shave
0.0952381	boys will shave themselves
0.047619	who will shave themselves

Problem #2 with the naive parametrization



Language of both grammars

boys will shave
boys will shave themselves
who will shave
who will shave themselves
some boys will shave
some boys will shave themselves

Training data

10	boys will shave
2	boys will shave themselves
3	who will shave
1	who will shave themselves
5	some boys will shave

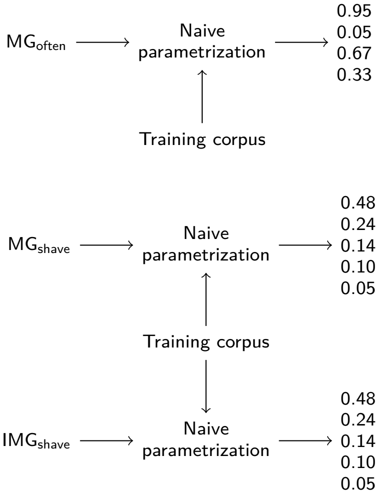
MG_{shave}, i.e. merge and move distinct	
0.47619	boys will shave
0.238095	some boys will shave
0.142857	who will shave
0.0952381	boys will shave themselves
0.047619	who will shave themselves

$\text{IMG}_{\text{shave}}$, i.e. merge and move unified	
0.47619	boys will shave
0.238095	some boys will shave
0.142857	who will shave
0.0952381	boys will shave themselves
0.047619	who will shave themselves

This treatment of probabilities doesn’t know which derivational operations are being implemented by the various MCFG rules.

So the probabilities are unaffected by changes in set of primitive operations.

Naive parametrization



Outline

The smarter parametrization

Solution: Have a rule's probability be a function of (only) "what it does"

- merge or move
- what feature is being checked (either movement or selection)

(Hunter and Dyer 2013)

The smarter parametrization

Solution: Have a rule's probability be a function of (only) "what it does"

- merge or move
- what feature is being checked (either movement or selection)

MCFG Rule	ϕ_{MERGE}	$\phi_{\mathtt{d}}$	$\phi_{\mathtt{v}}$	$\phi_{\mathtt{t}}$	ϕ_{MOVE}	$\phi_{\mathtt{wh}}$
$st :: \langle \mathtt{c} \rangle_0 \rightarrow s :: \langle \mathtt{=t\ c} \rangle_1 \quad t :: \langle \mathtt{t} \rangle_0$	1	0	0	1	0	0
$ts :: \langle \mathtt{c} \rangle_0 \rightarrow \langle s, t \rangle :: \langle \mathtt{+wh\ c}, \mathtt{-wh} \rangle_0$	0	0	0	0	1	1
$st :: \langle \mathtt{v} \rangle_0 \rightarrow s :: \langle \mathtt{=d\ v} \rangle_1 \quad t :: \langle \mathtt{d} \rangle_1$	1	1	0	0	0	0
$st :: \langle \mathtt{v} \rangle_0 \rightarrow s :: \langle \mathtt{=v\ v} \rangle_1 \quad t :: \langle \mathtt{v} \rangle_0$	1	0	1	0	0	0
$\langle s, t \rangle :: \langle \mathtt{v}, \mathtt{-wh} \rangle_0 \rightarrow s :: \langle \mathtt{=d\ v} \rangle_1 \quad t :: \langle \mathtt{d\ -wh} \rangle_1$	1	1	0	0	0	0
$\langle st, u \rangle :: \langle \mathtt{v}, \mathtt{-wh} \rangle_0 \rightarrow s :: \langle \mathtt{=v\ v} \rangle_1 \quad \langle t, u \rangle :: \langle \mathtt{v}, \mathtt{-wh} \rangle_0$	1	0	1	0	0	0

Each rule r is assigned a score as a function of the vector $\phi(r)$:

$$\begin{aligned} s(r) &= \exp(\boldsymbol{\lambda} \cdot \phi(r)) \\ &= \exp(\lambda_{\text{MERGE}}\, \phi_{\text{MERGE}}(r) + \lambda_{\mathtt{d}}\, \phi_{\mathtt{d}}(r) + \lambda_{\mathtt{v}}\, \phi_{\mathtt{v}}(r) + \dots) \end{aligned}$$

(Hunter and Dyer 2013)

The smarter parametrization

Solution: Have a rule's probability be a function of (only) "what it does"

- merge or move
- what feature is being checked (either movement or selection)

MCFG Rule	ϕ_{MERGE}	$\phi_{\mathtt{d}}$	$\phi_{\mathtt{v}}$	$\phi_{\mathtt{t}}$	ϕ_{MOVE}	$\phi_{\mathtt{wh}}$
$st :: \langle \mathtt{c} \rangle_0 \rightarrow s :: \langle \mathtt{=t\ c} \rangle_1 \quad t :: \langle \mathtt{t} \rangle_0$	1	0	0	1	0	0
$ts :: \langle \mathtt{c} \rangle_0 \rightarrow \langle s, t \rangle :: \langle \mathtt{+wh\ c}, \mathtt{-wh} \rangle_0$	0	0	0	0	1	1
$st :: \langle \mathtt{v} \rangle_0 \rightarrow s :: \langle \mathtt{=d\ v} \rangle_1 \quad t :: \langle \mathtt{d} \rangle_1$	1	1	0	0	0	0
$st :: \langle \mathtt{v} \rangle_0 \rightarrow s :: \langle \mathtt{=v\ v} \rangle_1 \quad t :: \langle \mathtt{v} \rangle_0$	1	0	1	0	0	0
$\langle s, t \rangle :: \langle \mathtt{v}, \mathtt{-wh} \rangle_0 \rightarrow s :: \langle \mathtt{=d\ v} \rangle_1 \quad t :: \langle \mathtt{d\ -wh} \rangle_1$	1	1	0	0	0	0
$\langle st, u \rangle :: \langle \mathtt{v}, \mathtt{-wh} \rangle_0 \rightarrow s :: \langle \mathtt{=v\ v} \rangle_1 \quad \langle t, u \rangle :: \langle \mathtt{v}, \mathtt{-wh} \rangle_0$	1	0	1	0	0	0

Each rule r is assigned a score as a function of the vector $\phi(r)$:

$$s(r) = \exp(\boldsymbol{\lambda} \cdot \phi(r))$$
$$= \exp(\lambda_{\text{MERGE}}\, \phi_{\text{MERGE}}(r) + \lambda_{\mathtt{d}}\, \phi_{\mathtt{d}}(r) + \lambda_{\mathtt{v}}\, \phi_{\mathtt{v}}(r) + \dots)$$

$$s(r_1) = \exp(\lambda_{\text{MERGE}} + \lambda_{\mathtt{t}})$$

(Hunter and Dyer 2013)

The smarter parametrization

Solution: Have a rule's probability be a function of (only) "what it does"

- merge or move
- what feature is being checked (either movement or selection)

MCFG Rule	ϕ_{MERGE}	$\phi_{\texttt{d}}$	$\phi_{\texttt{v}}$	$\phi_{\texttt{t}}$	ϕ_{MOVE}	$\phi_{\texttt{wh}}$
$st :: \langle \texttt{c} \rangle_0 \rightarrow s :: \langle \texttt{=t c} \rangle_1 \quad t :: \langle \texttt{t} \rangle_0$	1	0	0	1	0	0
$ts :: \langle \texttt{c} \rangle_0 \rightarrow \langle s, t \rangle :: \langle \texttt{+wh c, -wh} \rangle_0$	0	0	0	0	1	1
$st :: \langle \texttt{v} \rangle_0 \rightarrow s :: \langle \texttt{=d v} \rangle_1 \quad t :: \langle \texttt{d} \rangle_1$	1	1	0	0	0	0
$st :: \langle \texttt{v} \rangle_0 \rightarrow s :: \langle \texttt{=v v} \rangle_1 \quad t :: \langle \texttt{v} \rangle_0$	1	0	1	0	0	0
$\langle s, t \rangle :: \langle \texttt{v, -wh} \rangle_0 \rightarrow s :: \langle \texttt{=d v} \rangle_1 \quad t :: \langle \texttt{d -wh} \rangle_1$	1	1	0	0	0	0
$\langle st, u \rangle :: \langle \texttt{v, -wh} \rangle_0 \rightarrow s :: \langle \texttt{=v v} \rangle_1 \quad \langle t, u \rangle :: \langle \texttt{v, -wh} \rangle_0$	1	0	1	0	0	0

Each rule r is assigned a score as a function of the vector $\phi(r)$:

$$s(r) = \exp(\boldsymbol{\lambda} \cdot \phi(r))$$
$$= \exp(\lambda_{\text{MERGE}}\, \phi_{\text{MERGE}}(r) + \lambda_{\texttt{d}}\, \phi_{\texttt{d}}(r) + \lambda_{\texttt{v}}\, \phi_{\texttt{v}}(r) + \dots)$$

$$s(r_1) = \exp(\lambda_{\text{MERGE}} + \lambda_{\texttt{t}})$$
$$s(r_2) = \exp(\lambda_{\text{MOVE}} + \lambda_{\texttt{wh}})$$

(Hunter and Dyer 2013)

The smarter parametrization

Solution: Have a rule's probability be a function of (only) "what it does"

- merge or move
- what feature is being checked (either movement or selection)

MCFG Rule	ϕ_{MERGE}	$\phi_{\mathtt{d}}$	$\phi_{\mathtt{v}}$	$\phi_{\mathtt{t}}$	ϕ_{MOVE}	$\phi_{\mathtt{wh}}$
$st :: \langle \mathtt{c} \rangle_0 \rightarrow s :: \langle \mathtt{=t\ c} \rangle_1 \quad t :: \langle \mathtt{t} \rangle_0$	1	0	0	1	0	0
$ts :: \langle \mathtt{c} \rangle_0 \rightarrow \langle s, t \rangle :: \langle \mathtt{+wh\ c, -wh} \rangle_0$	0	0	0	0	1	1
$st :: \langle \mathtt{v} \rangle_0 \rightarrow s :: \langle \mathtt{=d\ v} \rangle_1 \quad t :: \langle \mathtt{d} \rangle_1$	1	1	0	0	0	0
$st :: \langle \mathtt{v} \rangle_0 \rightarrow s :: \langle \mathtt{=v\ v} \rangle_1 \quad t :: \langle \mathtt{v} \rangle_0$	1	0	1	0	0	0
$\langle s, t \rangle :: \langle \mathtt{v, -wh} \rangle_0 \rightarrow s :: \langle \mathtt{=d\ v} \rangle_1 \quad t :: \langle \mathtt{d\ -wh} \rangle_1$	1	1	0	0	0	0
$\langle st, u \rangle :: \langle \mathtt{v, -wh} \rangle_0 \rightarrow s :: \langle \mathtt{=v\ v} \rangle_1 \quad \langle t, u \rangle :: \langle \mathtt{v, -wh} \rangle_0$	1	0	1	0	0	0

Each rule r is assigned a score as a function of the vector $\phi(r)$:

$$s(r) = \exp(\boldsymbol{\lambda} \cdot \phi(r))$$
$$= \exp(\lambda_{\text{MERGE}}\, \phi_{\text{MERGE}}(r) + \lambda_{\mathtt{d}}\, \phi_{\mathtt{d}}(r) + \lambda_{\mathtt{v}}\, \phi_{\mathtt{v}}(r) + \dots)$$

$$s(r_1) = \exp(\lambda_{\text{MERGE}} + \lambda_{\mathtt{t}})$$
$$s(r_2) = \exp(\lambda_{\text{MOVE}} + \lambda_{\mathtt{wh}})$$
$$s(r_3) = \exp(\lambda_{\text{MERGE}} + \lambda_{\mathtt{d}})$$

(Hunter and Dyer 2013)

The smarter parametrization

Solution: Have a rule's probability be a function of (only) "what it does"

- merge or move
- what feature is being checked (either movement or selection)

MCFG Rule	ϕ_{MERGE}	$\phi_{\mathtt{d}}$	$\phi_{\mathtt{v}}$	$\phi_{\mathtt{t}}$	ϕ_{MOVE}	$\phi_{\mathtt{wh}}$
$st :: \langle \mathtt{c} \rangle_0 \rightarrow s :: \langle \mathtt{=t\ c} \rangle_1 \quad t :: \langle \mathtt{t} \rangle_0$	1	0	0	1	0	0
$ts :: \langle \mathtt{c} \rangle_0 \rightarrow \langle s, t \rangle :: \langle \mathtt{+wh\ c}, \mathtt{-wh} \rangle_0$	0	0	0	0	1	1
$st :: \langle \mathtt{v} \rangle_0 \rightarrow s :: \langle \mathtt{=d\ v} \rangle_1 \quad t :: \langle \mathtt{d} \rangle_1$	1	1	0	0	0	0
$st :: \langle \mathtt{v} \rangle_0 \rightarrow s :: \langle \mathtt{=v\ v} \rangle_1 \quad t :: \langle \mathtt{v} \rangle_0$	1	0	1	0	0	0
$\langle s, t \rangle :: \langle \mathtt{v}, \mathtt{-wh} \rangle_0 \rightarrow s :: \langle \mathtt{=d\ v} \rangle_1 \quad t :: \langle \mathtt{d\ -wh} \rangle_1$	1	1	0	0	0	0
$\langle st, u \rangle :: \langle \mathtt{v}, \mathtt{-wh} \rangle_0 \rightarrow s :: \langle \mathtt{=v\ v} \rangle_1 \quad \langle t, u \rangle :: \langle \mathtt{v}, \mathtt{-wh} \rangle_0$	1	0	1	0	0	0

Each rule r is assigned a score as a function of the vector $\phi(r)$:

$$s(r) = \exp(\boldsymbol{\lambda} \cdot \phi(r))$$

$$= \exp(\lambda_{\text{MERGE}}\, \phi_{\text{MERGE}}(r) + \lambda_{\mathtt{d}}\, \phi_{\mathtt{d}}(r) + \lambda_{\mathtt{v}}\, \phi_{\mathtt{v}}(r) + \dots)$$

$$s(r_1) = \exp(\lambda_{\text{MERGE}} + \lambda_{\mathtt{t}})$$

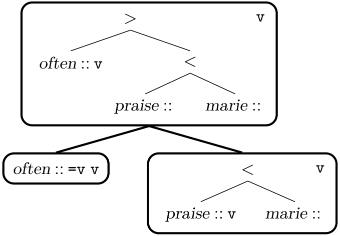
$$s(r_2) = \exp(\lambda_{\text{MOVE}} + \lambda_{\mathtt{wh}})$$

$$s(r_3) = \exp(\lambda_{\text{MERGE}} + \lambda_{\mathtt{d}})$$

$$s(r_5) = \exp(\lambda_{\text{MERGE}} + \lambda_{\mathtt{d}})$$

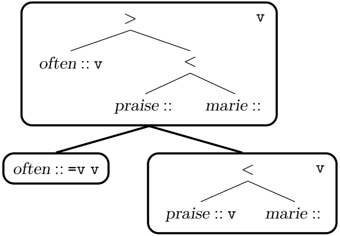
(Hunter and Dyer 2013)

Generalizations missed by the naive parametrization

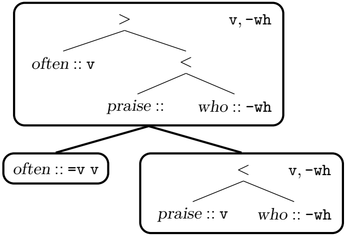


$$\textit{st} :: \langle \mathtt{v} \rangle_0 \quad \rightarrow \quad \textit{s} :: \langle \mathtt{=v\ v} \rangle_1 \quad \textit{t} :: \langle \mathtt{v} \rangle_0$$

Generalizations missed by the naive parametrization



$$st :: \langle \mathtt{v} \rangle_0 \quad \rightarrow \quad s :: \langle \mathtt{=v\ v} \rangle_1 \quad t :: \langle \mathtt{v} \rangle_0$$



$$\langle st, u \rangle :: \langle \mathtt{v}, \mathtt{-wh} \rangle_0 \quad \rightarrow \quad s :: \langle \mathtt{=v\ v} \rangle_1 \quad \langle t, u \rangle :: \langle \mathtt{v}, \mathtt{-wh} \rangle_0$$

Comparison

The old way:

λ_1	$ts :: \langle \texttt{c} \rangle_0$	$\rightarrow$	$\langle s, t \rangle :: \langle \texttt{+wh c}, \texttt{-wh} \rangle_0$
λ_2	$st :: \langle \texttt{c} \rangle_0$	$\rightarrow$	$s :: \langle \texttt{=t c} \rangle_1 \quad t :: \langle \texttt{t} \rangle_0$
λ_3	$st :: \langle \texttt{v} \rangle_0$	$\rightarrow$	$s :: \langle \texttt{=d v} \rangle_1 \quad t :: \langle \texttt{d} \rangle_1$
λ_4	$st :: \langle \texttt{v} \rangle_0$	$\rightarrow$	$s :: \langle \texttt{=v v} \rangle_1 \quad t :: \langle \texttt{v} \rangle_0$
λ_5	$\langle s, t \rangle :: \langle \texttt{v}, \texttt{-wh} \rangle_0$	$\rightarrow$	$s :: \langle \texttt{=d v} \rangle_1 \quad t :: \langle \texttt{d -wh} \rangle_1$
λ_6	$\langle st, u \rangle :: \langle \texttt{v}, \texttt{-wh} \rangle_0$	$\rightarrow$	$s :: \langle \texttt{=v v} \rangle_1 \quad \langle t, u \rangle :: \langle \texttt{v}, \texttt{-wh} \rangle_0$

Training question: What values of λ_1, λ_2, etc. make the training corpus most likely?

The new way:

$\exp(\lambda_{\text{MOVE}} + \lambda_{\texttt{wh}})$	$ts :: \langle \texttt{c} \rangle_0$	$\rightarrow$	$\langle s, t \rangle :: \langle \texttt{+wh c}, \texttt{-wh} \rangle_0$
$\exp(\lambda_{\text{MERGE}} + \lambda_{\texttt{t}})$	$st :: \langle \texttt{c} \rangle_0$	$\rightarrow$	$s :: \langle \texttt{=t c} \rangle_1 \quad t :: \langle \texttt{t} \rangle_0$
$\exp(\lambda_{\text{MERGE}} + \lambda_{\texttt{d}})$	$st :: \langle \texttt{v} \rangle_0$	$\rightarrow$	$s :: \langle \texttt{=d v} \rangle_1 \quad t :: \langle \texttt{d} \rangle_1$
$\exp(\lambda_{\text{MERGE}} + \lambda_{\texttt{v}})$	$st :: \langle \texttt{v} \rangle_0$	$\rightarrow$	$s :: \langle \texttt{=v v} \rangle_1 \quad t :: \langle \texttt{v} \rangle_0$
$\exp(\lambda_{\text{MERGE}} + \lambda_{\texttt{d}})$	$\langle s, t \rangle :: \langle \texttt{v}, \texttt{-wh} \rangle_0$	$\rightarrow$	$s :: \langle \texttt{=d v} \rangle_1 \quad t :: \langle \texttt{d -wh} \rangle_1$
$\exp(\lambda_{\text{MERGE}} + \lambda_{\texttt{v}})$	$\langle st, u \rangle :: \langle \texttt{v}, \texttt{-wh} \rangle_0$	$\rightarrow$	$s :: \langle \texttt{=v v} \rangle_1 \quad \langle t, u \rangle :: \langle \texttt{v}, \texttt{-wh} \rangle_0$

Training question: What values of λ_{MERGE}, λ_{MOVE}, $\lambda_{\texttt{d}}$, etc. make the training corpus most likely?

Solution #1 with the smarter parametrization

Grammar

pierre :: `d`	*who* :: `d -wh`
marie :: `d`	*will* :: `=v =d t`
praise :: `=d v`	ϵ :: `=t c`
often :: `=v v`	ϵ :: `=t +wh c`

Training data

90	pierre will praise marie
5	pierre will **often** praise marie
1	who pierre will praise
1	who pierre will **often** praise

Maximise likelihood via stochastic gradient ascent:

$$P_{\boldsymbol{\lambda}}(N \to \delta) = \frac{\exp(\boldsymbol{\lambda} \cdot \phi(N \to \delta))}{\sum \exp(\boldsymbol{\lambda} \cdot \phi(N \to \delta'))}$$

Solution #1 with the smarter parametrization

Grammar

pierre :: d	*who* :: d -wh
marie :: d	*will* :: =v =d t
praise :: =d v	ϵ :: =t c
often :: =v v	ϵ :: =t +wh c

Training data

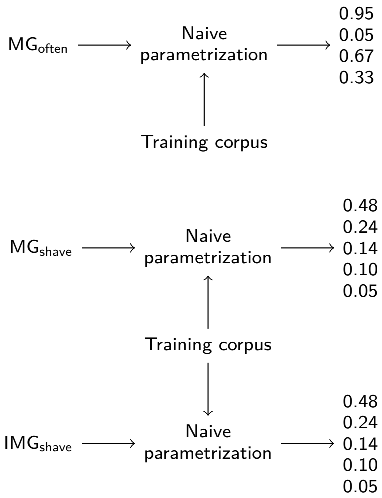
90	pierre will praise marie
5	pierre will **often** praise marie
1	who pierre will praise
1	who pierre will **often** praise

Maximise likelihood via stochastic gradient ascent:

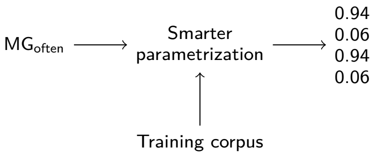
$$P_{\boldsymbol{\lambda}}(N \to \delta) = \frac{\exp(\boldsymbol{\lambda} \cdot \phi(N \to \delta))}{\sum \exp(\boldsymbol{\lambda} \cdot \phi(N \to \delta'))}$$

				naive	smarter
$st :: \langle \mathtt{v} \rangle_0$	$\to$	$s :: \langle \mathtt{=d\ v} \rangle_1$	$t :: \langle \mathtt{d} \rangle_1$	0.95	0.94
$st :: \langle \mathtt{v} \rangle_0$	$\to$	$s :: \langle \mathtt{=v\ v} \rangle_1$	$t :: \langle \mathtt{v} \rangle_0$	0.05	0.06
$\langle s, t \rangle :: \langle \mathtt{v}, \mathtt{-wh} \rangle_0$	$\to$	$s :: \langle \mathtt{=d\ v} \rangle_1$	$t :: \langle \mathtt{d\ -wh} \rangle_1$	**0.67**	**0.94**
$\langle st, u \rangle :: \langle \mathtt{v}, \mathtt{-wh} \rangle_0$	$\to$	$s :: \langle \mathtt{=v\ v} \rangle_1$	$\langle t, u \rangle :: \langle \mathtt{v}, \mathtt{-wh} \rangle_0$	**0.33**	**0.06**

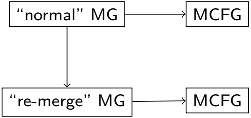
Naive parametrization



Smarter parametrization



Solution #2 with the smarter parametrization



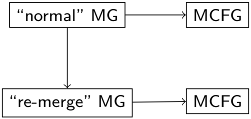
Language of both grammars

boys will shave
boys will shave themselves
who will shave
who will shave themselves
some boys will shave
some boys will shave themselves

Training data

10	boys will shave
2	boys will shave themselves
3	who will shave
1	who will shave themselves
5	some boys will shave

Solution #2 with the smarter parametrization



Language of both grammars

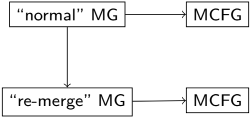
boys will shave
boys will shave themselves
who will shave
who will shave themselves
some boys will shave
some boys will shave themselves

Training data

10	boys will shave
2	boys will shave themselves
3	who will shave
1	who will shave themselves
5	some boys will shave

MG_{shave}, i.e. merge and move distinct	
0.35478	boys will shave
0.35478	some boys will shave
0.14801	who will shave
0.05022	boys will shave themselves
0.05022	some boys will shave themselves
0.04199	who will shave themselves

Solution #2 with the smarter parametrization



Language of both grammars

boys will shave
boys will shave themselves
who will shave
who will shave themselves
some boys will shave
some boys will shave themselves

Training data

10	boys will shave
2	boys will shave themselves
3	who will shave
1	who will shave themselves
5	some boys will shave

MG_{shave}, i.e. merge and move distinct	
0.35478	boys will shave
0.35478	some boys will shave
0.14801	who will shave
0.05022	boys will shave themselves
0.05022	some boys will shave themselves
0.04199	who will shave themselves

IMG_{shave}, i.e. merge and move unified	
0.35721	boys will shave
0.35721	some boys will shave
0.095	who will shave
0.095	who will shave themselves
0.04779	boys will shave themselves
0.04779	some boys will shave themselves

Naive parametrization

MG_{often} → Naive parametrization → 0.95, 0.05, 0.67, 0.33
(Training corpus → Naive parametrization)

MG_{shave} → Naive parametrization → 0.48, 0.24, 0.14, 0.10, 0.05
(Training corpus → Naive parametrization)

IMG_{shave} → Naive parametrization → 0.48, 0.24, 0.14, 0.10, 0.05
(Training corpus → Naive parametrization)

Smarter parametrization

MG_{often} → Smarter parametrization → 0.94, 0.06, 0.94, 0.06
(Training corpus → Smarter parametrization)

MG_{shave} → Smarter parametrization → 0.35, 0.35, 0.15, 0.05, 0.05, 0.04
(Training corpus → Smarter parametrization)

IMG_{shave} → Smarter parametrization → 0.36, 0.36, 0.10, 0.10, 0.05, 0.05
(Training corpus → Smarter parametrization)

Choice points in the MG-derived MCFG

Question or not?			
$\langle \texttt{c} \rangle_0$	$\rightarrow$	$\langle \texttt{=t c} \rangle_0$	$\langle \texttt{t} \rangle_0$
$\langle \texttt{c} \rangle_0$	$\rightarrow$	$\langle \texttt{+wh c}, \texttt{-wh} \rangle_0$	

Antecedent lexical or complex?			
$\langle \texttt{ant -subj} \rangle_0$	$\rightarrow$	$\langle \texttt{=subj ant -subj} \rangle_1$	$\langle \texttt{subj} \rangle_0$
$\langle \texttt{ant -subj} \rangle_0$	$\rightarrow$	$\langle \texttt{=subj ant -subj} \rangle_1$	$\langle \texttt{subj} \rangle_1$

Non-wh subject merged and complex, merged and lexical, or moved?			
$\langle \texttt{t} \rangle_0$	$\rightarrow$	$\langle \texttt{=subj t} \rangle_0$	$\langle \texttt{subj} \rangle_0$
$\langle \texttt{t} \rangle_0$	$\rightarrow$	$\langle \texttt{=subj t} \rangle_0$	$\langle \texttt{subj} \rangle_1$
$\langle \texttt{t} \rangle_0$	$\rightarrow$	$\langle \texttt{+subj t}, \texttt{-subj} \rangle_0$	

Wh-phrase same as moving subject or separated because of doubling?			
$\langle \texttt{t}, \texttt{-wh} \rangle_0$	$\rightarrow$	$\langle \texttt{=subj t} \rangle_0$	$\langle \texttt{subj -wh} \rangle_1$
$\langle \texttt{t}, \texttt{-wh} \rangle_0$	$\rightarrow$	$\langle \texttt{+subj t}, \texttt{-subj}, \texttt{-wh} \rangle_0$	

Choice points in the MG-derived MCFG

Question or not?				
$\langle \mathtt{c} \rangle_0$	$\rightarrow$	$\langle \mathtt{=t\ c} \rangle_0$	$\langle \mathtt{t} \rangle_0$	$\exp(\lambda_{\text{MERGE}} + \lambda_{\mathtt{t}})$
$\langle \mathtt{c} \rangle_0$	$\rightarrow$	$\langle \mathtt{+wh\ c, -wh} \rangle_0$		$\exp(\lambda_{\text{MOVE}} + \lambda_{\mathtt{wh}})$

Antecedent lexical or complex?				
$\langle \mathtt{ant\ -subj} \rangle_0$	$\rightarrow$	$\langle \mathtt{=subj\ ant\ -subj} \rangle_1$	$\langle \mathtt{subj} \rangle_0$	$\exp(\lambda_{\text{MERGE}} + \lambda_{\mathtt{subj}})$
$\langle \mathtt{ant\ -subj} \rangle_0$	$\rightarrow$	$\langle \mathtt{=subj\ ant\ -subj} \rangle_1$	$\langle \mathtt{subj} \rangle_1$	$\exp(\lambda_{\text{MERGE}} + \lambda_{\mathtt{subj}})$

Non-wh subject merged and complex, merged and lexical, or moved?				
$\langle \mathtt{t} \rangle_0$	$\rightarrow$	$\langle \mathtt{=subj\ t} \rangle_0$	$\langle \mathtt{subj} \rangle_0$	$\exp(\lambda_{\text{MERGE}} + \lambda_{\mathtt{subj}})$
$\langle \mathtt{t} \rangle_0$	$\rightarrow$	$\langle \mathtt{=subj\ t} \rangle_0$	$\langle \mathtt{subj} \rangle_1$	$\exp(\lambda_{\text{MERGE}} + \lambda_{\mathtt{subj}})$
$\langle \mathtt{t} \rangle_0$	$\rightarrow$	$\langle \mathtt{+subj\ t, -subj} \rangle_0$		$\exp(\lambda_{\text{MOVE}} + \lambda_{\mathtt{subj}})$

Wh-phrase same as moving subject or separated because of doubling?				
$\langle \mathtt{t, -wh} \rangle_0$	$\rightarrow$	$\langle \mathtt{=subj\ t} \rangle_0$	$\langle \mathtt{subj\ -wh} \rangle_1$	$\exp(\lambda_{\text{MERGE}} + \lambda_{\mathtt{subj}})$
$\langle \mathtt{t, -wh} \rangle_0$	$\rightarrow$	$\langle \mathtt{+subj\ t, -subj, -wh} \rangle_0$		$\exp(\lambda_{\text{MOVE}} + \lambda_{\mathtt{subj}})$

Choice points in the IMG-derived MCFG

Question or not?		
$\langle\texttt{-c}\rangle_0$	$\to$	$\langle\texttt{+t -c}, \texttt{-t}\rangle_1$
$\langle\texttt{-c}\rangle_0$	$\to$	$\langle\texttt{+wh -c}, \texttt{-wh}\rangle_0$

Antecedent lexical or complex?			
$\langle\texttt{+subj -ant -subj}, \texttt{-subj}\rangle_0$	$\to$	$\langle\texttt{+subj -ant -subj}\rangle_0$	$\langle\texttt{-subj}\rangle_0$
$\langle\texttt{+subj -ant -subj}, \texttt{-subj}\rangle_0$	$\to$	$\langle\texttt{+subj -ant -subj}\rangle_0$	$\langle\texttt{-subj}\rangle_1$

Non-wh subject merged and complex, merged and lexical, or moved?			
$\langle\texttt{+subj -t}, \texttt{-subj}\rangle_0$	$\to$	$\langle\texttt{+subj -t}\rangle_0$	$\langle\texttt{-subj}\rangle_0$
$\langle\texttt{+subj -t}, \texttt{-subj}\rangle_0$	$\to$	$\langle\texttt{+subj -t}\rangle_0$	$\langle\texttt{-subj}\rangle_1$
$\langle\texttt{+subj -t}, \texttt{-subj}\rangle_0$	$\to$	$\langle\texttt{+v +subj -t}, \texttt{-v}, \texttt{-subj}\rangle_1$	

Wh-phrase same as moving subject or separated because of doubling?		
$\langle\texttt{-t}, \texttt{-wh}\rangle_0$	$\to$	$\langle\texttt{+subj -t}, \texttt{-subj -wh}\rangle_0$
$\langle\texttt{-t}, \texttt{-wh}\rangle_0$	$\to$	$\langle\texttt{+subj -t}, \texttt{-subj}, \texttt{-wh}\rangle_0$

Choice points in the IMG-derived MCFG

Question or not?

$$\langle \texttt{-c} \rangle_0 \rightarrow \langle \texttt{+t -c}, \texttt{-t} \rangle_1 \qquad \exp(\lambda_{\text{MRG}} + \lambda_{\texttt{t}})$$
$$\langle \texttt{-c} \rangle_0 \rightarrow \langle \texttt{+wh -c}, \texttt{-wh} \rangle_0 \qquad \exp(\lambda_{\text{MRG}} + \lambda_{\texttt{wh}})$$

Antecedent lexical or complex?

$$\langle \texttt{+subj -ant -subj}, \texttt{-subj} \rangle_0 \rightarrow \langle \texttt{+subj -ant -subj} \rangle_0 \quad \langle \texttt{-subj} \rangle_0 \qquad \exp(\lambda_{\text{INSERT}})$$
$$\langle \texttt{+subj -ant -subj}, \texttt{-subj} \rangle_0 \rightarrow \langle \texttt{+subj -ant -subj} \rangle_0 \quad \langle \texttt{-subj} \rangle_1 \qquad \exp(\lambda_{\text{INSERT}})$$

Non-wh subject merged and complex, merged and lexical, or moved?

$$\langle \texttt{+subj -t}, \texttt{-subj} \rangle_0 \rightarrow \langle \texttt{+subj -t} \rangle_0 \quad \langle \texttt{-subj} \rangle_0 \qquad \exp(\lambda_{\text{INSERT}})$$
$$\langle \texttt{+subj -t}, \texttt{-subj} \rangle_0 \rightarrow \langle \texttt{+subj -t} \rangle_0 \quad \langle \texttt{-subj} \rangle_1 \qquad \exp(\lambda_{\text{INSERT}})$$
$$\langle \texttt{+subj -t}, \texttt{-subj} \rangle_0 \rightarrow \langle \texttt{+v +subj -t}, \texttt{-v}, \texttt{-subj} \rangle_1 \qquad \exp(\lambda_{\text{MRG}} + \lambda_{\texttt{v}})$$

Wh-phrase same as moving subject or separated because of doubling?

$$\langle \texttt{-t}, \texttt{-wh} \rangle_0 \rightarrow \langle \texttt{+subj -t}, \texttt{-subj -wh} \rangle_0 \qquad \exp(\lambda_{\text{MRG}} + \lambda_{\texttt{subj}})$$
$$\langle \texttt{-t}, \texttt{-wh} \rangle_0 \rightarrow \langle \texttt{+subj -t}, \texttt{-subj}, \texttt{-wh} \rangle_0 \qquad \exp(\lambda_{\text{MRG}} + \lambda_{\texttt{subj}})$$

Learned weights on the MG

$$
\begin{aligned}
\lambda_{\texttt{t}} &= 0.094350 & \text{exp}(\lambda_{\texttt{t}}) &= 1.0989 \\
\lambda_{\texttt{subj}} &= -5.734063 & \text{exp}(\lambda_{\texttt{v}}) &= 0.0032 \\
\lambda_{\texttt{wh}} &= -0.094350 & \text{exp}(\lambda_{\texttt{wh}}) &= 0.9100 \\
\lambda_{\text{MERGE}} &= 0.629109 & \text{exp}(\lambda_{\text{MERGE}}) &= 1.8759 \\
\lambda_{\text{MOVE}} &= -0.629109 & \text{exp}(\lambda_{\text{MOVE}}) &= 0.5331
\end{aligned}
$$

Learned weights on the MG

$$\lambda_{\texttt{t}} = 0.094350 \qquad \exp(\lambda_{\texttt{t}}) = 1.0989$$
$$\lambda_{\texttt{subj}} = -5.734063 \qquad \exp(\lambda_{\texttt{v}}) = 0.0032$$
$$\lambda_{\texttt{wh}} = -0.094350 \qquad \exp(\lambda_{\texttt{wh}}) = 0.9100$$
$$\lambda_{\text{MERGE}} = 0.629109 \qquad \exp(\lambda_{\text{MERGE}}) = 1.8759$$
$$\lambda_{\text{MOVE}} = -0.629109 \qquad \exp(\lambda_{\text{MOVE}}) = 0.5331$$

$$P(\text{antecedent is lexical}) = 0.5$$
$$P(\text{antecedent is non-lexical}) = 0.5$$

$$P(\text{wh-phrase reflexivized}) = \frac{\exp(\lambda_{\text{MOVE}})}{\exp(\lambda_{\text{MERGE}}) + \exp(\lambda_{\text{MOVE}})} = 0.2213$$
$$P(\text{wh-phrase non-reflexivized}) = \frac{\exp(\lambda_{\text{MERGE}})}{\exp(\lambda_{\text{MERGE}}) + \exp(\lambda_{\text{MOVE}})} = 0.7787$$

$$P(\text{question}) = \frac{\exp(\lambda_{\text{MOVE}} + \lambda_{\texttt{wh}})}{\exp(\lambda_{\text{MERGE}} + \lambda_{\texttt{t}}) + \exp(\lambda_{\text{MOVE}} + \lambda_{\texttt{wh}})} = 0.1905$$
$$P(\text{non-question}) = \frac{\exp(\lambda_{\text{MERGE}} + \lambda_{\texttt{t}})}{\exp(\lambda_{\text{MERGE}} + \lambda_{\texttt{t}}) + \exp(\lambda_{\text{MOVE}} + \lambda_{\texttt{wh}})} = 0.8095$$

$$P(\text{non-wh subject merged and complex}) = \frac{\exp(\lambda_{\text{MERGE}})}{\exp(\lambda_{\text{MERGE}}) + \exp(\lambda_{\text{MERGE}}) + \exp(\lambda_{\text{MOVE}})} = 0.4378$$
$$P(\text{non-wh subject merged and lexical}) = \frac{\exp(\lambda_{\text{MERGE}})}{\exp(\lambda_{\text{MERGE}}) + \exp(\lambda_{\text{MERGE}}) + \exp(\lambda_{\text{MOVE}})} = 0.4378$$
$$P(\text{non-wh subject moved}) = \frac{\exp(\lambda_{\text{MOVE}})}{\exp(\lambda_{\text{MERGE}}) + \exp(\lambda_{\text{MERGE}}) + \exp(\lambda_{\text{MOVE}})} = 0.1244$$

Learned weights on the MG

$$P(\text{antecedent is lexical}) = 0.5$$
$$P(\text{antecedent is non-lexical}) = 0.5$$

$$\lambda_{\texttt{t}} = 0.094350 \qquad \exp(\lambda_{\texttt{t}}) = 1.0989$$
$$\lambda_{\texttt{subj}} = -5.734063 \qquad \exp(\lambda_{\texttt{v}}) = 0.0032$$
$$\lambda_{\texttt{wh}} = -0.094350 \qquad \exp(\lambda_{\texttt{wh}}) = 0.9100$$
$$\lambda_{\text{MERGE}} = 0.629109 \qquad \exp(\lambda_{\text{MERGE}}) = 1.8759$$
$$\lambda_{\text{MOVE}} = -0.629109 \qquad \exp(\lambda_{\text{MOVE}}) = 0.5331$$

$$P(\text{wh-phrase reflexivized}) = \frac{\exp(\lambda_{\text{MOVE}})}{\exp(\lambda_{\text{MERGE}}) + \exp(\lambda_{\text{MOVE}})} = 0.2213$$
$$P(\text{wh-phrase non-reflexivized}) = \frac{\exp(\lambda_{\text{MERGE}})}{\exp(\lambda_{\text{MERGE}}) + \exp(\lambda_{\text{MOVE}})} = 0.7787$$

$$P(\text{question}) = \frac{\exp(\lambda_{\text{MOVE}} + \lambda_{\texttt{wh}})}{\exp(\lambda_{\text{MERGE}} + \lambda_{\texttt{t}}) + \exp(\lambda_{\text{MOVE}} + \lambda_{\texttt{wh}})} = 0.1905$$
$$P(\text{non-question}) = \frac{\exp(\lambda_{\text{MERGE}} + \lambda_{\texttt{t}})}{\exp(\lambda_{\text{MERGE}} + \lambda_{\texttt{t}}) + \exp(\lambda_{\text{MOVE}} + \lambda_{\texttt{wh}})} = 0.8095$$

$$P(\text{non-wh subject merged and complex}) = \frac{\exp(\lambda_{\text{MERGE}})}{\exp(\lambda_{\text{MERGE}}) + \exp(\lambda_{\text{MERGE}}) + \exp(\lambda_{\text{MOVE}})} = 0.4378$$
$$P(\text{non-wh subject merged and lexical}) = \frac{\exp(\lambda_{\text{MERGE}})}{\exp(\lambda_{\text{MERGE}}) + \exp(\lambda_{\text{MERGE}}) + \exp(\lambda_{\text{MOVE}})} = 0.4378$$
$$P(\text{non-wh subject moved}) = \frac{\exp(\lambda_{\text{MOVE}})}{\exp(\lambda_{\text{MERGE}}) + \exp(\lambda_{\text{MERGE}}) + \exp(\lambda_{\text{MOVE}})} = 0.1244$$

$$P(\text{who will shave}) = 0.1905 \times 0.7787 = 0.148$$
$$P(\text{boys will shave themselves}) = 0.5 \times 0.8095 \times 0.1244 = 0.050$$

Learned weights on the IMG

$$
\begin{aligned}
\lambda_{\texttt{t}} &= 0.723549 & \exp(\lambda_{\texttt{t}}) &= 2.0617 \\
\lambda_{\texttt{v}} &= 0.440585 & \exp(\lambda_{\texttt{v}}) &= 1.5536 \\
\lambda_{\texttt{wh}} &= -0.723459 & \exp(\lambda_{\texttt{wh}}) &= 0.4850 \\
\lambda_{\text{INSERT}} &= 0.440585 & \exp(\lambda_{\text{INSERT}}) &= 1.5536 \\
\lambda_{\text{MRG}} &= -0.440585 & \exp(\lambda_{\text{MRG}}) &= 0.6437
\end{aligned}
$$

Learned weights on the IMG

$$\lambda_{\texttt{t}} = 0.723549 \qquad \exp(\lambda_{\texttt{t}}) = 2.0617$$
$$\lambda_{\texttt{v}} = 0.440585 \qquad \exp(\lambda_{\texttt{v}}) = 1.5536$$
$$\lambda_{\texttt{wh}} = -0.723459 \qquad \exp(\lambda_{\texttt{wh}}) = 0.4850$$
$$\lambda_{\text{INSERT}} = 0.440585 \qquad \exp(\lambda_{\text{INSERT}}) = 1.5536$$
$$\lambda_{\text{MRG}} = -0.440585 \qquad \exp(\lambda_{\text{MRG}}) = 0.6437$$

$$P(\text{antecedent is lexical}) = 0.5$$
$$P(\text{antecedent is non-lexical}) = 0.5$$

$$P(\text{wh-phrase reflexivized}) = 0.5$$
$$P(\text{wh-phrase non-reflexivized}) = 0.5$$

$$P(\text{question}) = \frac{\exp(\lambda_{\text{MRG}} + \lambda_{\texttt{wh}})}{\exp(\lambda_{\text{MRG}} + \lambda_{\texttt{t}}) + \exp(\lambda_{\text{MRG}} + \lambda_{\texttt{wh}})} = \frac{\exp(\lambda_{\texttt{wh}})}{\exp(\lambda_{\texttt{t}}) + \exp(\lambda_{\texttt{wh}})} = 0.1905$$
$$P(\text{non-question}) = \frac{\exp(\lambda_{\text{MRG}} + \lambda_{\texttt{t}})}{\exp(\lambda_{\text{MRG}} + \lambda_{\texttt{t}}) + \exp(\lambda_{\text{MRG}} + \lambda_{\texttt{wh}})} = \frac{\exp(\lambda_{\texttt{t}})}{\exp(\lambda_{\texttt{t}}) + \exp(\lambda_{\texttt{wh}})} = 0.8095$$

$$P(\text{non-wh subject merged and lexical}) = \frac{\exp(\lambda_{\text{INSERT}})}{\exp(\lambda_{\text{INSERT}}) + \exp(\lambda_{\text{INSERT}}) + \exp(\lambda_{\text{MRG}} + \lambda_{\texttt{v}})} = 0.4412$$
$$P(\text{non-wh subject merged and complex}) = \frac{\exp(\lambda_{\text{INSERT}})}{\exp(\lambda_{\text{INSERT}}) + \exp(\lambda_{\text{INSERT}}) + \exp(\lambda_{\text{MRG}} + \lambda_{\texttt{v}})} = 0.4412$$
$$P(\text{non-wh subject moved}) = \frac{\exp(\lambda_{\text{MRG}} + \lambda_{\texttt{v}})}{\exp(\lambda_{\text{INSERT}}) + \exp(\lambda_{\text{INSERT}}) + \exp(\lambda_{\text{MRG}} + \lambda_{\texttt{v}})} = 0.1176$$

Learned weights on the IMG

$$\lambda_{\texttt{t}} = 0.723549 \qquad \exp(\lambda_{\texttt{t}}) = 2.0617$$
$$\lambda_{\texttt{v}} = 0.440585 \qquad \exp(\lambda_{\texttt{v}}) = 1.5536$$
$$\lambda_{\texttt{wh}} = -0.723459 \qquad \exp(\lambda_{\texttt{wh}}) = 0.4850$$
$$\lambda_{\text{INSERT}} = 0.440585 \qquad \exp(\lambda_{\text{INSERT}}) = 1.5536$$
$$\lambda_{\text{MRG}} = -0.440585 \qquad \exp(\lambda_{\text{MRG}}) = 0.6437$$

$$P(\text{antecedent is lexical}) = 0.5$$
$$P(\text{antecedent is non-lexical}) = 0.5$$

$$P(\text{wh-phrase reflexivized}) = 0.5$$
$$P(\text{wh-phrase non-reflexivized}) = 0.5$$

$$P(\text{question}) = \frac{\exp(\lambda_{\text{MRG}} + \lambda_{\texttt{wh}})}{\exp(\lambda_{\text{MRG}} + \lambda_{\texttt{t}}) + \exp(\lambda_{\text{MRG}} + \lambda_{\texttt{wh}})} = \frac{\exp(\lambda_{\texttt{wh}})}{\exp(\lambda_{\texttt{t}}) + \exp(\lambda_{\texttt{wh}})} = 0.1905$$
$$P(\text{non-question}) = \frac{\exp(\lambda_{\text{MRG}} + \lambda_{\texttt{t}})}{\exp(\lambda_{\text{MRG}} + \lambda_{\texttt{t}}) + \exp(\lambda_{\text{MRG}} + \lambda_{\texttt{wh}})} = \frac{\exp(\lambda_{\texttt{t}})}{\exp(\lambda_{\texttt{t}}) + \exp(\lambda_{\texttt{wh}})} = 0.8095$$

$$P(\text{non-wh subject merged and lexical}) = \frac{\exp(\lambda_{\text{INSERT}})}{\exp(\lambda_{\text{INSERT}}) + \exp(\lambda_{\text{INSERT}}) + \exp(\lambda_{\text{MRG}} + \lambda_{\texttt{v}})} = 0.4412$$
$$P(\text{non-wh subject merged and complex}) = \frac{\exp(\lambda_{\text{INSERT}})}{\exp(\lambda_{\text{INSERT}}) + \exp(\lambda_{\text{INSERT}}) + \exp(\lambda_{\text{MRG}} + \lambda_{\texttt{v}})} = 0.4412$$
$$P(\text{non-wh subject moved}) = \frac{\exp(\lambda_{\text{MRG}} + \lambda_{\texttt{v}})}{\exp(\lambda_{\text{INSERT}}) + \exp(\lambda_{\text{INSERT}}) + \exp(\lambda_{\text{MRG}} + \lambda_{\texttt{v}})} = 0.1176$$

$$P(\text{who will shave}) = 0.5 \times 0.1905 = 0.095$$
$$P(\text{boys will shave themselves}) = 0.5 \times 0.8095 \times 0.1176 = 0.048$$

Surprisal predictions

Grammar: MG_{shave}
Sentence: ‘who will shave themselves’

MG_{shave}, i.e. merge and move distinct	
0.35478	boys will shave
0.35478	some boys will shave
0.14801	who will shave
0.05022	boys will shave themselves
0.05022	some boys will shave themselves
0.04199	who will shave themselves

Surprisal predictions

Grammar: $\mathrm{MG}_{\mathrm{shave}}$
Sentence: 'who will shave themselves'

$\mathrm{MG}_{\mathrm{shave}}$, i.e. merge and move distinct	
0.35478	boys will shave
0.35478	some boys will shave
0.14801	who will shave
0.05022	boys will shave themselves
0.05022	some boys will shave themselves
0.04199	who will shave themselves

$$\begin{aligned}
\text{surprisal at 'who'} &= -\log P(W_1 = \text{who}) \\
&= -\log(0.15 + 0.04) \\
&= -\log 0.19 \\
&= 2.4 \\
\text{surprisal at 'themselves'} &= -\log P(W_4 = \text{themselves} \mid W_1 = \text{who}, \ldots) \\
&= -\log \frac{0.04}{0.15 + 0.04} \\
&= -\log 0.21 \\
&= 2.2
\end{aligned}$$

Surprisal predictions

Grammar: $\text{IMG}_{\text{shave}}$
Sentence: 'who will shave themselves'

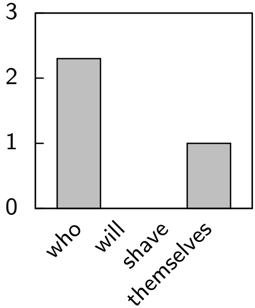
$\text{IMG}_{\text{shave}}$, i.e. merge and move unified	
0.35721	boys will shave
0.35721	some boys will shave
0.095	who will shave
0.095	who will shave themselves
0.04779	boys will shave themselves
0.04779	some boys will shave themselves

Surprisal predictions

Grammar: $\text{IMG}_{\text{shave}}$
Sentence: ‘who will shave themselves’

$\text{IMG}_{\text{shave}}$, i.e. merge and move unified	
0.35721	boys will shave
0.35721	some boys will shave
0.095	who will shave
0.095	who will shave themselves
0.04779	boys will shave themselves
0.04779	some boys will shave themselves

$$\begin{aligned}
\text{surprisal at ‘who’} &= -\log P(W_1 = \text{who}) \\
&= -\log(0.10 + 0.10) \\
&= -\log 0.2 \\
&= 2.3 \\
\text{surprisal at ‘themselves’} &= -\log P(W_4 = \text{themselves} \mid W_1 = \text{who}, \dots) \\
&= -\log \frac{0.10}{0.10 + 0.10} \\
&= -\log 0.5 \\
&= 1
\end{aligned}$$



Sharpening the empirical claims of generative syntax
through formalization

Tim Hunter — ESSLLI, August 2015

Part 5

Learning and wrap-up

Motivating question

Components of a learner:

- A formalism ("toolkit") defines a space of grammars for a learner to choose from
- An updating algorithm defines a way to search through such a space (in response to provided input)

Motivating question

Components of a learner:

- A formalism (“toolkit”) defines a space of grammars for a learner to choose from
- An updating algorithm defines a way to search through such a space (in response to provided input)

Given two formalisms, F1 and F2, can we construct a learner which

- reaches one end-state when used with F1, and
- reaches a different end-state when used with F2?

Motivating question

Components of a learner:

- A formalism ("toolkit") defines a space of grammars for a learner to choose from
- An updating algorithm defines a way to search through such a space (in response to provided input)

Given two formalisms, F1 and F2, can we construct a learner which

- reaches one end-state when used with F1, and
- reaches a different end-state when used with F2?

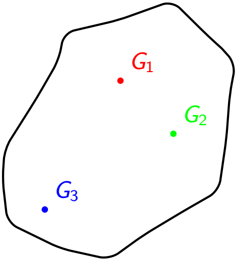
With everything else held fixed:

- same (strong) generative capacity
- same updating algorithm
- same training data

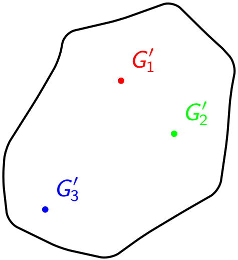
Outline

Outline

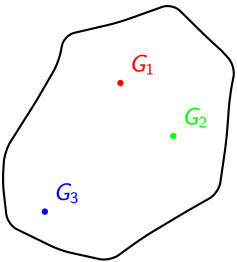
Formalism F1



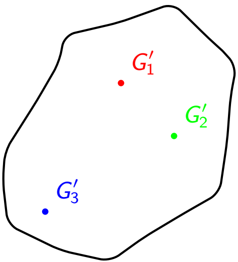
Formalism F2



Formalism F1



Formalism F2

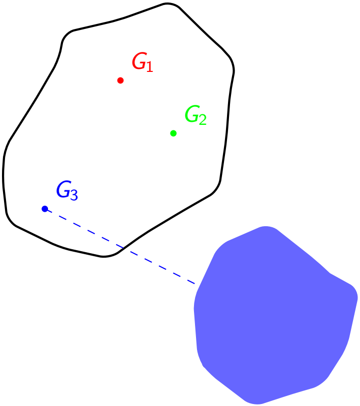


A "good sentence vs. bad sentence" learner will treat these two formalisms equivalently — it won't "see" the internal differences in how they generate what they generate.

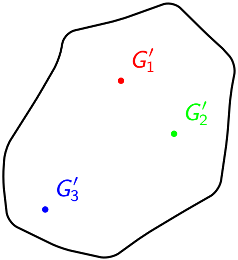
(Gibson and Wexler 1994)

Q: How can we provide traction between the learning algorithm and the internals of each G?

Formalism F1



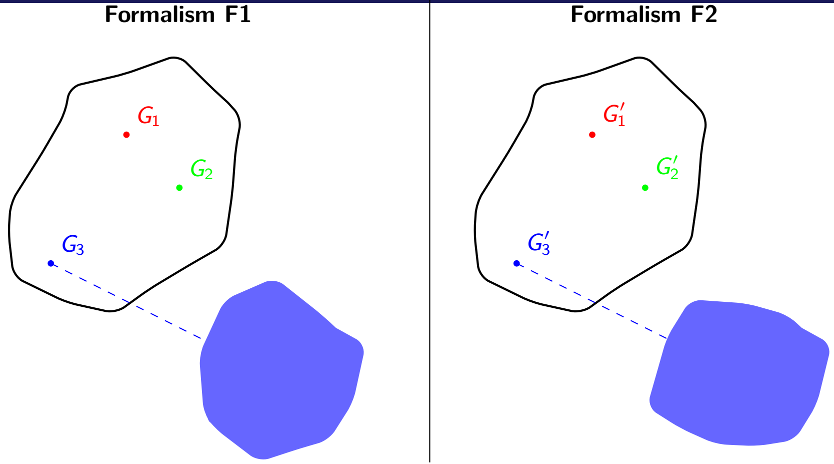
Formalism F2



A "good sentence vs. bad sentence" learner will treat these two formalisms equivalently — it won't "see" the internal differences in how they generate what they generate.

(Gibson and Wexler 1994)

Q: How can we provide traction between the learning algorithm and the internals of each G?
A: Probabilities



A "good sentence vs. bad sentence" learner will treat these two formalisms equivalently — it won't "see" the internal differences in how they generate what they generate.

(Gibson and Wexler 1994)

Q: How can we provide traction between the learning algorithm and the internals of each G?
A: Probabilities

Outline

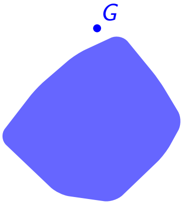
Learning scenario

Training corpus: some combination of occurrences of the following.

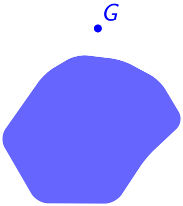
boys will shave	boys will shave themselves
who will shave	who will shave themselves
foo boys will shave	

- The learner knows correct analyses of these sentences, with ‘foo’ as a determiner.
- The learner must decide what probabilities to attach to these known sentences.

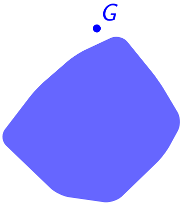
MGs



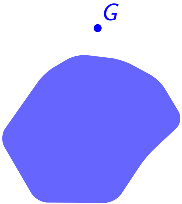
IMGs



MGs



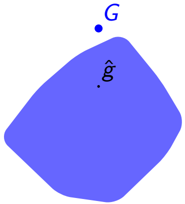
IMGs



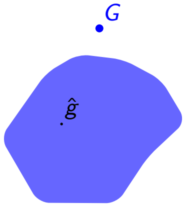
Training corpus:

10	boys will shave
2	boys will shave themselves
3	who will shave
1	who will shave themselves
5	foo boys will shave

MGs



IMGs



Training corpus:

10	boys will shave
2	boys will shave themselves
3	who will shave
1	who will shave themselves
5	foo boys will shave

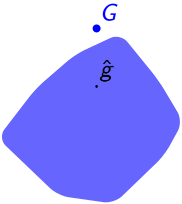
Grammar's distribution:

0.35478	boys will shave
0.35478	foo boys will shave
0.14801	who will shave
0.05022	boys will shave themselves
0.05022	foo boys will shave themselves
0.04199	who will shave themselves

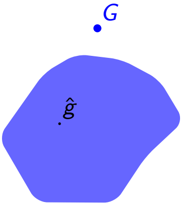
Grammar's distribution:

0.35721	boys will shave
0.35721	foo boys will shave
0.095	who will shave
0.095	who will shave themselves
0.04779	boys will shave themselves
0.04779	foo boys will shave themselves

MGs



IMGs



Training corpus:

10	boys will shave
2	boys will shave themselves
3	who will shave
1	who will shave themselves
5	foo boys will shave

	Entropy	Entropy Reduction
—	2.09	—
who	0.76	1.33
will	0.76	0.00
shave	0.76	0.00
themselves	0.00	0.76

	Entropy	Entropy Reduction
—	2.28	—
who	1.00	1.28
will	1.00	0.00
shave	1.00	0.00
themselves	0.00	1.00

Outline

Learning scenario

Training corpus: some combination of occurrences of the following.

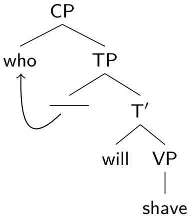
boys will shave	boys will shave themselves
who will shave	who will shave themselves
foo boys will shave	

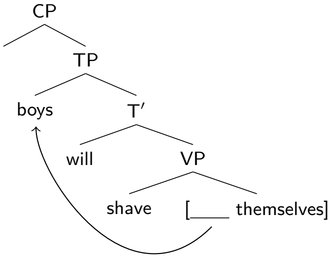
Learning scenario

Training corpus: some combination of occurrences of the following.

boys will shave	boys will shave themselves
who will shave	who will shave themselves
foo boys will shave	

- The learner knows correct analyses of wh-movement and reflexives.



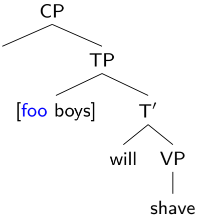


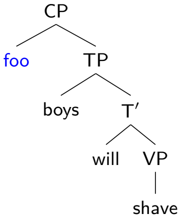
Learning scenario

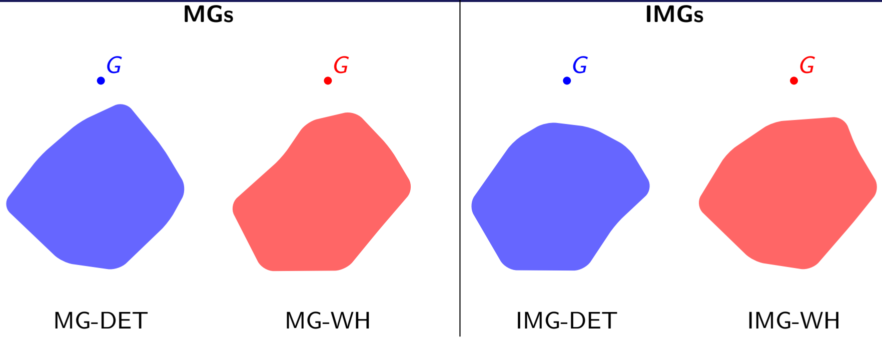
Training corpus: some combination of occurrences of the following.

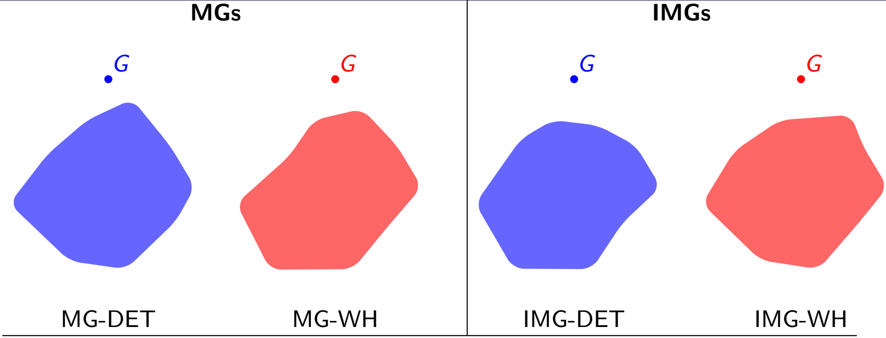
boys will shave	boys will shave themselves
who will shave	who will shave themselves
foo boys will shave	

- The learner knows correct analyses of wh-movement and reflexives.
- The learner must decide how to analyze ‘foo’: determiner or wh-phrase?



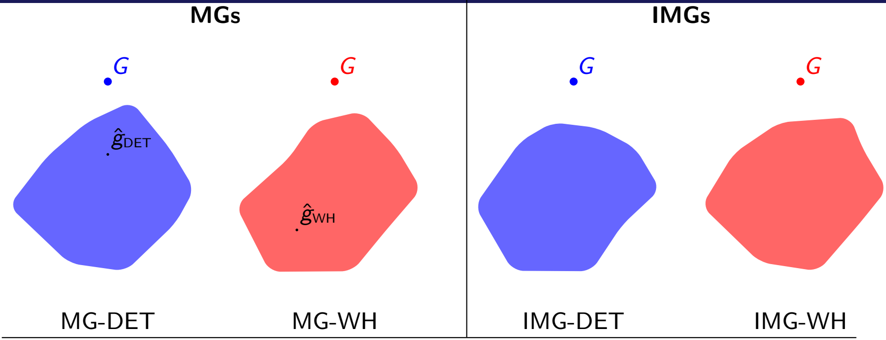



MGs
IMGs
G
G
G
G
MG-DET
MG-WH
IMG-DET
IMG-WH



Training corpus:

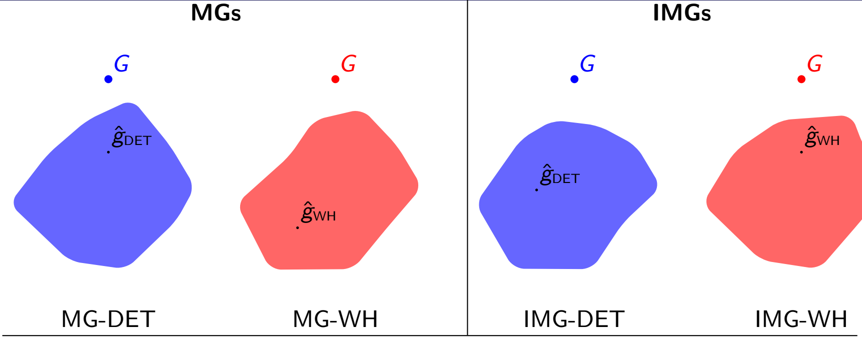
5	boys will shave
5	boys will shave themselves
5	who will shave
5	who will shave themselves
5	foo boys will shave



Training corpus:

5	boys will shave
5	boys will shave themselves
5	who will shave
5	who will shave themselves
5	foo boys will shave

$$\frac{P(D|\hat{g}_{\text{DET}})}{P(D|\hat{g}_{\text{WH}})} = \frac{3.36 \times 10^{-18}}{4.48 \times 10^{-20}} = 75.0$$

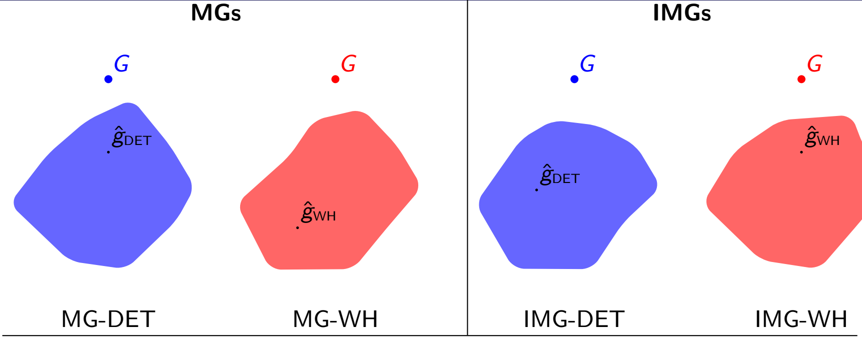


Training corpus:

5	boys will shave
5	boys will shave themselves
5	who will shave
5	who will shave themselves
5	foo boys will shave

$$\frac{P(D|\hat{g}_{\text{DET}})}{P(D|\hat{g}_{\text{WH}})} = \frac{3.36 \times 10^{-18}}{4.48 \times 10^{-20}} = 75.0$$

$$\frac{P(D|\hat{g}_{\text{DET}})}{P(D|\hat{g}_{\text{WH}})} = \frac{3.36 \times 10^{-18}}{2.45 \times 10^{-19}} = 13.7$$

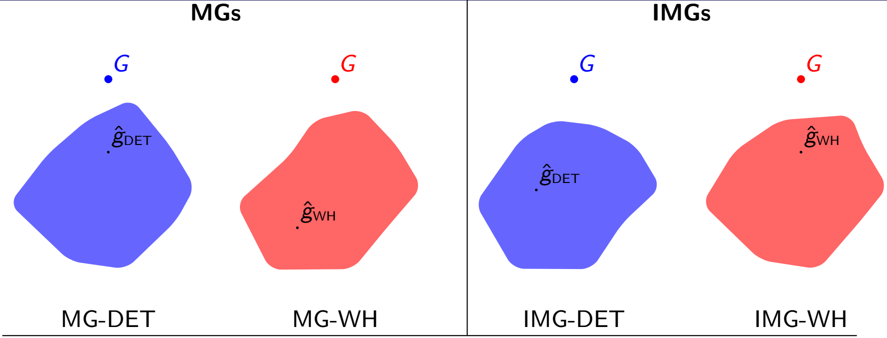


Training corpus:

18	boys will shave
3	boys will shave themselves
1	who will shave
1	who will shave themselves
1	foo boys will shave

$$\frac{P(D|\hat{g}_{\text{DET}})}{P(D|\hat{g}_{\text{WH}})} = \frac{5.82 \times 10^{-14}}{7.27 \times 10^{-11}} = 0.000801$$

$$\frac{P(D|\hat{g}_{\text{DET}})}{P(D|\hat{g}_{\text{WH}})} = \frac{7.64 \times 10^{-14}}{6.85 \times 10^{-10}} = 0.000112$$

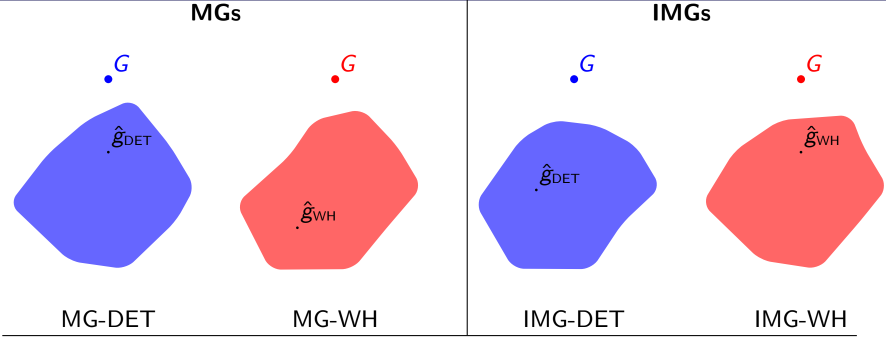


Training corpus:

1	boys will shave
1	boys will shave themselves
8	who will shave
8	who will shave themselves
8	foo boys will shave

$$\frac{P(D|\hat{g}_{\text{DET}})}{P(D|\hat{g}_{\text{WH}})} = \frac{1.21 \times 10^{-17}}{7.70 \times 10^{-19}} = 15.7$$

$$\frac{P(D|\hat{g}_{\text{DET}})}{P(D|\hat{g}_{\text{WH}})} = \frac{3.46 \times 10^{-17}}{1.19 \times 10^{-16}} = 0.291$$

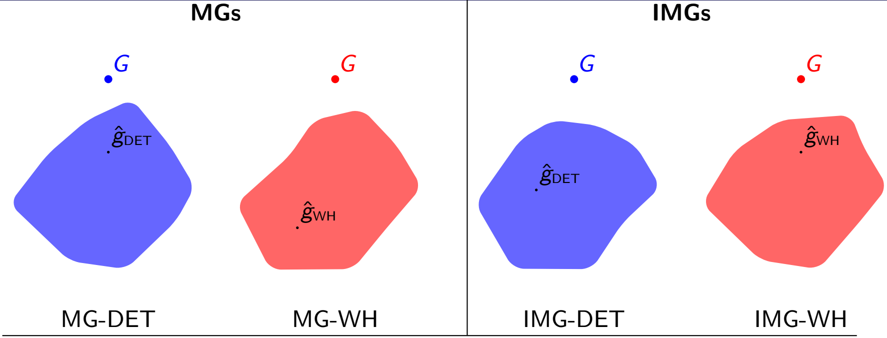


Training corpus:

8	boys will shave
1	boys will shave themselves
12	who will shave
1	who will shave themselves
4	foo boys will shave

$$\frac{P(D|\hat{g}_{\text{DET}})}{P(D|\hat{g}_{\text{WH}})} = \frac{2.83 \times 10^{-15}}{4.36 \times 10^{-20}} = 64900$$

$$\frac{P(D|\hat{g}_{\text{DET}})}{P(D|\hat{g}_{\text{WH}})} = \frac{1.31 \times 10^{-17}}{1.75 \times 10^{-17}} = 0.749$$



Training corpus:

10	boys will shave
2	boys will shave themselves
3	who will shave
1	who will shave themselves
5	foo boys will shave

$$\frac{P(D|\hat{g}_{\text{DET}})}{P(D|\hat{g}_{\text{WH}})} = \frac{2.44 \times 10^{-13}}{4.94 \times 10^{-14}} = 4.94$$

$$\frac{P(D|\hat{g}_{\text{DET}})}{P(D|\hat{g}_{\text{WH}})} = \frac{1.46 \times 10^{-13}}{1.62 \times 10^{-13}} = 0.901$$

Details of one interesting case

MG-WH

```
Feature weight: ant=0.000000
Feature weight: obj=0.000000
Feature weight: subj=0.306077
Feature weight: t=-0.895880
Feature weight: v=0.000000
Feature weight: wh=0.895880
Feature weight: merge=-0.000000
Feature weight: move=-0.000000
{t29: 0.5, t13_t4: 0.5}
{t28: 0.5, t13_t5: 0.5}
{t0_t14: 0.077, t21_t7: 0.462, t22: 0.462}
```

```
t0 : (:: =t c)
t4 : (:: subj)
t5 : (:: subj -wh)
t7 : (:: wh)
t13 : (: =subj t)
t14 : (: t)
t21 : (: =wh c)
t22 : (: +wh c;: -wh)
t28 : (: +subj t;: -subj;: -wh)
t29 : (: +subj t;: -subj)
```

IMG-WH

```
Feature weight: ant=0.000000
Feature weight: obj=0.000000
Feature weight: subj=-0.860545
Feature weight: t=-0.434630
Feature weight: v=-3.324996
Feature weight: wh=2.050275
Feature weight: insert=-0.563888
Feature weight: merge=0.563888
{t00130005: 0.5, t0028: 0.5}
{t0021_t0007: 0.333, t00010016: 0.667}
{t00000014: 0.077, t0022: 0.923}
{t0013_t0004: 0.900, t00110026: 0.100}
```

```
t00000014 : (:: +t -c;: -t)
t00010016 : (:: +t +wh -c;: -t;: -wh)
t0004 : (:: -subj)
t0007 : (:: -wh)
t00110026 : (:: +v +subj -t;: -v;: -subj)
t0013 : (: +subj -t)
t00130005 : (: +subj -t;: -subj -wh)
t0021 : (: +wh -c)
t0022 : (: +wh -c;: -wh)
t0028 : (: +subj -t;: -subj;: -wh)
```

Outline

What we've done (I hope)

If we accept — as I do — . . . that the rules of grammar enter into the processing mechanisms, then evidence concerning production, recognition, recall, and language use in general can be expected (in principle) to have bearing on the investigation of rules of grammar, on what is sometimes called "grammatical competence" or "knowledge of language".

(Chomsky 1980: pp.200-201)

The psychological plausibility of a transformational model of the language user would be strengthened, of course, if it could be shown that our performance on tasks requiring an appreciation of the structure of transformed sentences is some function of the nature, number and complexity of the grammatical transformations involved.

(Miller and Chomsky 1963: p.481)

What we've done (I hope)

There are ways to have "purely derivational" properties of formalisms make a difference to predictions about sentence processing complexity and generalization in learning

What we've done (I hope)

There are ways to have "purely derivational" properties of formalisms make a difference to predictions about sentence processing complexity and generalization in learning

- . . . without saying anything about real-time mental operations
- . . . (let alone saying that things like MERGE and MOVE happen in real time).

What we've done (I hope)

There are ways to have "purely derivational" properties of formalisms make a difference to predictions about sentence processing complexity and generalization in learning

- ... without saying anything about real-time mental operations
- ... (let alone saying that things like MERGE and MOVE happen in real time).
- Instead, the derivation tree is the object to be recovered/identified.

What we've done (I hope)

There are ways to have "purely derivational" properties of formalisms make a difference to predictions about sentence processing complexity and generalization in learning

- . . . without saying anything about real-time mental operations
- . . . (let alone saying that things like MERGE and MOVE happen in real time).
- Instead, the derivation tree is the object to be recovered/identified.

> *As mentioned above, the MP as a syntactic theory appears to be a step backwards for psycholinguistics (although perhaps not for syntacticians, of course). One of the fundamental problems is that the model derives a tree starting from all the lexical items and working up to the top-most node, which obviously is difficult to reconcile with left-to-right incremental parsing*
>
> *Ferreira (2005: p.369)*

What we've done (I hope)

There are ways to have "purely derivational" properties of formalisms make a difference to predictions about sentence processing complexity and generalization in learning

- . . . without saying anything about real-time mental operations
- . . . (let alone saying that things like MERGE and MOVE happen in real time).
- Instead, the derivation tree is the object to be recovered/identified.

> *As mentioned above, the MP as a syntactic theory appears to be a step backwards for psycholinguistics (although perhaps not for syntacticians, of course). One of the fundamental problems is that the model derives a tree starting from all the lexical items and working up to the top-most node, which obviously is difficult to reconcile with left-to-right incremental parsing*
>
> *Ferreira (2005: p.369)*

- What we've done of course leaves questions about real-time operations unanswered.
- But it's not clear that there is a conflict that needs to be "reconciled".

Open questions

How realistic is the assumption that there are a finite number of derivational states?

- MGs' SMC vs. mainstream "minimality"
- Dependencies over arbitrary distances (e.g. Condition C, NPIs)
- . . . ?

Local vs. global normalization

References I

Billot, S. and Lang, B. (1989). The structure of shared forests in ambiguous parsing. In *Proceedings of the 1989 Meeting of the Association of Computational Linguistics*.

Chomsky, N. (1965). *Aspects of the Theory of Syntax*. MIT Press, Cambridge, MA.

Chomsky, N. (1980). *Rules and Representations*. Columbia University Press, New York.

Ferreira, F. (2005). Psycholinguistics, formal grammars, and cognitive science. *The Linguistic Review*, 22:365–380.

Frazier, L. and Clifton, C. (1996). *Construal*. MIT Press, Cambridge, MA.

Gärtner, H.-M. and Michaelis, J. (2010). On the Treatment of Multiple-Wh Interrogatives in Minimalist Grammars. In Hanneforth, T. and Fanselow, G., editors, *Language and Logos*, pages 339–366. Akademie Verlag, Berlin.

Gibson, E. and Wexler, K. (1994). Triggers. *Linguistic Inquiry*, 25:407–454.

Hale, J. (2006). Uncertainty about the rest of the sentence. *Cognitive Science*, 30:643–Âŋ672.

Hale, J. T. (2001). A probabilistic earley parser as a psycholinguistic model. In *Proceedings of the Second Meeting of the North American Chapter of the Association for Computational Linguistics*.

Hunter, T. (2011). Insertion Minimalist Grammars: Eliminating redundancies between merge and move. In Kanazawa, M., Kornai, A., Kracht, M., and Seki, H., editors, *The Mathematics of Language (MOL 12 Proceedings)*, volume 6878 of *LNCS*, pages 90–107, Berlin Heidelberg. Springer.

Hunter, T. and Dyer, C. (2013). Distributions on minimalist grammar derivations. In *Proceedings of the 13th Meeting on the Mathematics of Language*.

Koopman, H. and Szabolcsi, A. (2000). *Verbal Complexes*. MIT Press, Cambridge, MA.

Lang, B. (1988). Parsing incomplete sentences. In *Proceedings of the 12th International Conference on Computational Linguistics*, pages 365–371.

Levy, R. (2008). Expectation-based syntactic comprehension. *Cognition*, 106(3):1126–1177.

Michaelis, J. (2001). Derivational minimalism is mildly context-sensitive. In Moortgat, M., editor, *Logical Aspects of Computational Linguistics*, volume 2014 of *LNCS*, pages 179–198. Springer, Berlin Heidelberg.

Miller, G. A. and Chomsky, N. (1963). Finitary models of language users. In Luce, R. D., Bush, R. R., and Galanter, E., editors, *Handbook of Mathematical Psychology*, volume 2. Wiley and Sons, New York.

Morrill, G. (1994). *Type Logical Grammar: Categorial Logic of Signs*. Kluwer, Dordrecht.

Nederhof, M. J. and Satta, G. (2008). Computing partition functions of pcfgs. *Research on Language and Computation*, 6(2):139–162.

Seki, H., Matsumara, T., Fujii, M., and Kasami, T. (1991). On multiple context-free grammars. *Theoretical Computer Science*, 88:191–229.

Stabler, E. P. (2006). Sidewards without copying. In Wintner, S., editor, *Proceedings of The 11th Conference on Formal Grammar*, pages 157–170, Stanford, CA. CSLI Publications.

Stabler, E. P. (2011). Computational perspectives on minimalism. In Boeckx, C., editor, *The Oxford Handbook of Linguistic Minimalism*. Oxford University Press, Oxford.

Stabler, E. P. and Keenan, E. L. (2003). Structural similarity within and among languages. *Theoretical Computer Science*, 293:345–363.

References III

Vijay-Shanker, K., Weir, D. J., and Joshi, A. K. (1987). Characterizing structural descriptions produced by various grammatical formalisms. In *Proceedings of the 25th Meeting of the Association for Computational Linguistics*, pages 104–111.

Weir, D. (1988). *Characterizing mildly context-sensitive grammar formalisms*. PhD thesis, University of Pennsylvania.

Yngve, V. H. (1960). A model and an hypothesis for language structure. In *Proceedings of the American Philosophical Society*, volume 104, pages 444–466.