# LING 510: Week 5 review

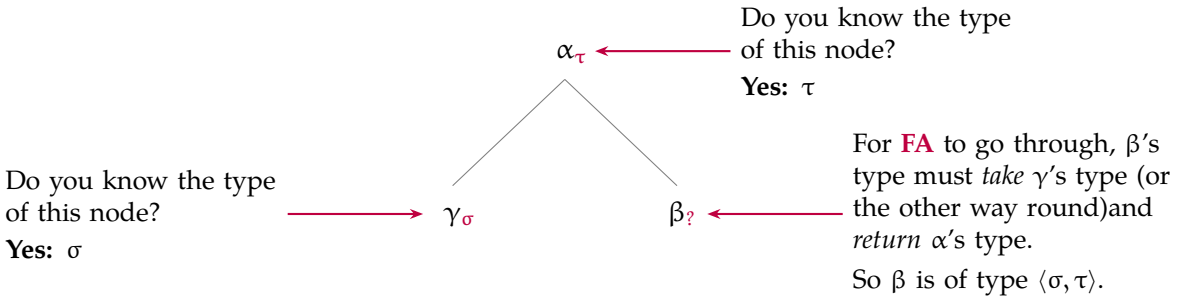
10/02/2026

Shaunak Phadnis
sphadnis@umass.edu
Office hours: W 11–1PM, N421H | Zoom

Where are we?

- Principle of compositionality and Syntax
- Type driven system of interpretation that assigns denotations to syntactic objects.
- Via rules of interpretation (Lexicon, TN, NN, FA)
- Turns out composition is just function application.

We have an inventory of types. Given a tree, we can guess what type a particular constituent should have based on its syntactic positions and the rule of function application. Here's a simple way to do it.




Remember that in a tree with multiple nodes with unknown types, you always know that a sentence is type t., names are type e, that intransitives are type $\langle e, t\rangle$. For all other nodes, you can infer their types recursively by the method described above.

Apart from that we also observed that

Natural language conjunction[a] is *flexible*. It can conjoin different kinds of constituents (as long as they are the same type!).

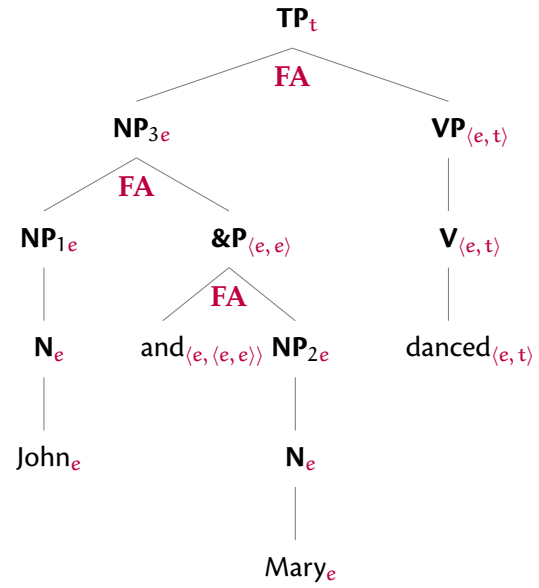
- Is there a separate entry for $\text{and}_{NP}$, an $\text{and}_{VP}$, an $\text{and}_{S}$, an $\text{and}_{AdvP}$,etc. except that all of them are spoken the same way as *'and'*?
- If so, then every time we discover new types of constituents that can be conjoined, we would need to add a new entry for conjunction corresponding to that type in the lexicon.
- But all of these conjunction do the same thing: they take two constituents of the same type and return a constituent of that same type!

Is that good design? Could we do better?

[a] Also disjunction

Here's a demonstration.

(1) John and Mary danced. NP-conjunction[1]



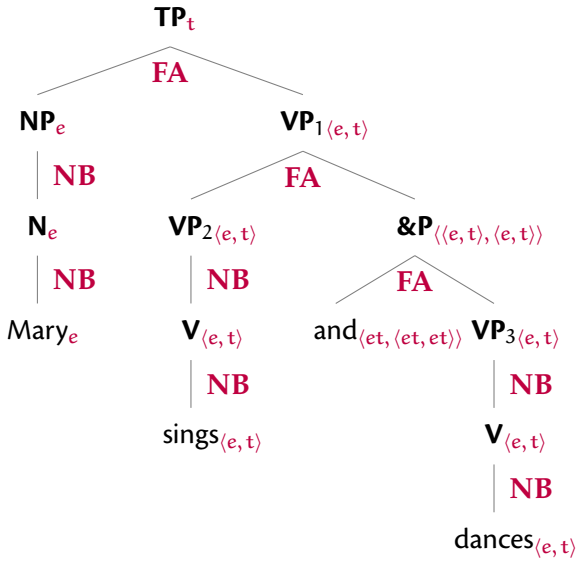

**Lexical entries**

$$[\![\text{John}]\!] = j$$
$$[\![\text{Mary}]\!] = m$$
$$[\![\text{and}]\!] = [\lambda x : x \in D_e . [\lambda y : y \in D_e . y \text{ and } x]]$$
$$[\![\text{danced}]\!] = [\lambda x : x \in D_e . T \text{ iff } x \text{ danced}]$$

**Composition**

$$[\![\&P]\!] = [\lambda y : y \in D_e . y \text{ and } m] \quad \textbf{FA}$$
$$[\![NP_3]\!] = j \text{ and } m \quad \textbf{FA}$$
$$[\![VP]\!] = [\lambda x : x \in D_e . T \text{ iff } x \text{ danced}] \quad \textbf{NB}$$
$$[\![TP]\!] = T \text{ iff } j \text{ and } m \text{ danced} \quad \textbf{FA}$$

(2) Mary sings and dances. VP-conjunction




**Lexical entries**

$$[\![\text{Mary}]\!] = m$$
$$[\![\text{sings}]\!] = [\lambda x : x \in D_e . T \text{ iff } x \text{ sings}]$$
$$[\![\text{dances}]\!] = [\lambda x : x \in D_e . T \text{ iff } x \text{ dances}]$$
$$[\![\text{and}]\!] = [\lambda f : f \in D_{\langle e,t\rangle} . [\lambda g : g \in D_{\langle e,t\rangle} . [\lambda x : x \in D_e . T \text{ iff } g(x) = T \text{ and } f(x) = T]]]$$

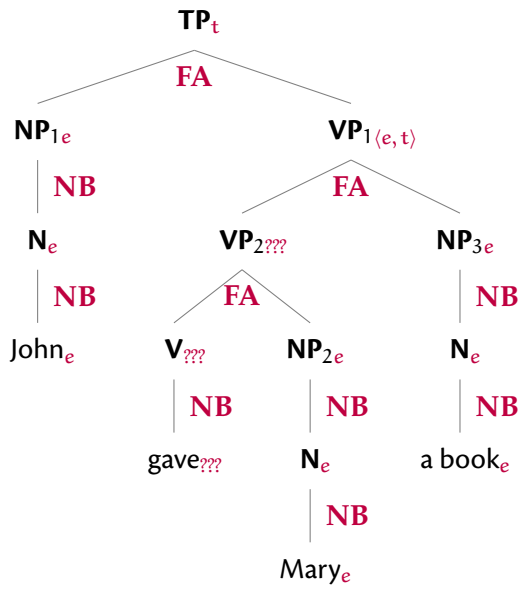
**Composition**

$$[\![\&P]\!] = [\lambda g : g \in D_{\langle e,t\rangle} . [\lambda x : x \in D_e . T \text{ iff } g(x) = T \text{ and } x \text{ dances}]] \quad \textbf{FA}$$
$$[\![VP_1]\!] = [\lambda x : x \in D_e . T \text{ iff } x \text{ sings and } x \text{ dances}] \quad \textbf{FA}$$
$$[\![TP]\!] = T \text{ iff } m \text{ sings and } m \text{ dances} \quad \textbf{FA}$$

[1] As we will see next week, there are issues with assigning $\langle e, \langle e, e\rangle\rangle$ type to NP-conjunction.

Last week we also figured out the type for the ditransitive verb *give*. Tell me the type again and figure out the right lexical entry that yields the correct truth conditions.

(3) John gave Mary a book.




**Lexical entries**

$[\![\text{John}]\!] = \text{j}$
$[\![\text{Mary}]\!] = \text{m}$
$[\![\text{a book}]\!] = \text{b}$
$[\![\text{gave}]\!] = ???$

**Composition**

$[\![\text{VP}_2]\!] = ???$
$[\![\text{VP}_1]\!] = ??$
$[\![\text{TP}]\!] = ?$

- John gave Mary a book is true just in case John was the agent of giving, a book was the thing being given, and Mary was the recipient.
- In order to write the lexical entry for gave, we need to capture this three-place relation between the agent (John), the beneficiary (Mary), and the theme (a book).
- But note that the internal argument of gave *does not* merge first in this structure.
- So when we write the lexical entry, the order of variables introduced should respect this order of composition.

Which of the following is the correct entry for gave? what would the output be for each?

1. $[\![\text{gave}]\!] = [\lambda x_e.[\lambda y_e.[\lambda z_e.\,\text{T iff } x \text{ gave } y \text{ to } z]]]$
2. $[\![\text{gave}]\!] = [\lambda x_e.[\lambda y_e.[\lambda z_e.\,\text{T iff } z \text{ gave } y \text{ to } x]]]$
3. $[\![\text{gave}]\!] = [\lambda x_e.[\lambda y_e.[\lambda z_e.\,\text{T iff } y \text{ gave } x \text{ to } z]]]$

**Subproof a.** $\mathrm{VP}_2$

1. $\left[\!\left[ \begin{array}{c} \mathbf{VP}_{2\langle e,\langle e,t\rangle\rangle} \\ \mathbf{V}_{\langle e,\langle e,\langle e,t\rangle\rangle\rangle} \quad \mathbf{NP}_{2e} \\ \mathbf{NB} \quad\quad \mathbf{NB} \\ \text{gave}_{\langle e,\langle e,\langle e,t\rangle\rangle\rangle} \quad \text{Mary}_e \end{array} \right]\!\right] = \left[\!\left[ \begin{array}{c} \mathbf{V}_{\langle e,\langle e,\langle e,t\rangle\rangle\rangle} \\ \mathbf{NB} \\ \text{gave}_{\langle e,\langle e,\langle e,t\rangle\rangle\rangle} \end{array} \right]\!\right] \left( \left[\!\left[ \begin{array}{c} \mathbf{NP}_{2e} \\ \mathbf{NB} \\ \text{Mary}_e \end{array} \right]\!\right] \right)$ (by FA)

2. $= [\![\text{gave}]\!]([\![\text{Mary}]\!])$ (by NB)

3. $= [\lambda x_e.[\lambda y_e.[\lambda z_e.\,\mathsf{T} \text{ iff } z \text{ gave } y \text{ to } x]]](\mathrm{m})$ (by lexical entries)

4. $= [\lambda y_e.[\lambda z_e.\,\mathsf{T} \text{ iff } z \text{ gave } y \text{ to m}]]$ (by β-reduction)

**Subproof b.** $\mathrm{VP}_1$

1. $\left[\!\left[ \begin{array}{c} \mathbf{VP}_{1\langle e,t\rangle} \\ \mathbf{VP}_{2\langle e,\langle e,t\rangle\rangle} \quad \mathbf{NP}_{3e} \\ \text{gave Mary} \quad\quad \text{a book} \end{array} \right]\!\right] = \left[\!\left[ \begin{array}{c} \mathbf{VP}_{2\langle e,\langle e,t\rangle\rangle} \\ \text{gave Mary} \end{array} \right]\!\right] \left( \left[\!\left[ \begin{array}{c} \mathbf{NP}_{3e} \\ \text{a book} \end{array} \right]\!\right] \right)$ (by FA)

2. $= [\lambda y_e.[\lambda z_e.\,\mathsf{T} \text{ iff } z \text{ gave } y \text{ to m}]](\mathrm{b})$ (by a., NB)

3. $= [\lambda z_e.\,\mathsf{T} \text{ iff } z \text{ gave b to m}]$ (by β-reduction)

**Derivation.** TP

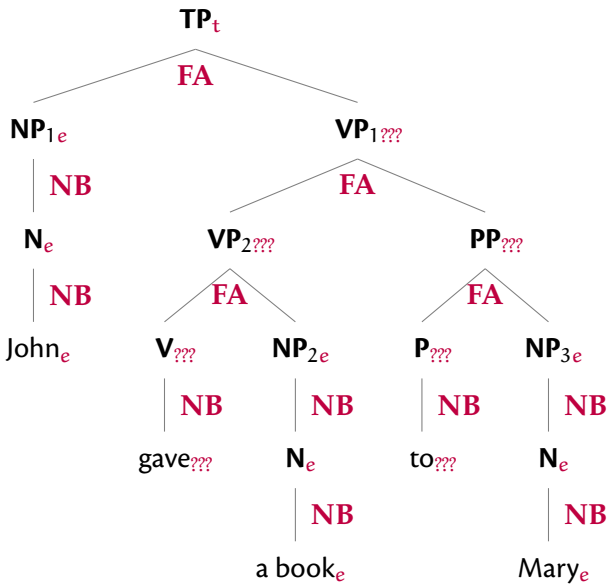
1. $\left[\!\left[ \begin{array}{c} \mathbf{TP}_t \\ \mathbf{NP}_{1e} \quad\quad \mathbf{VP}_{1\langle e,t\rangle} \\ \text{John} \quad\quad \text{gave Mary a book} \end{array} \right]\!\right] = \left[\!\left[ \begin{array}{c} \mathbf{VP}_{1\langle e,t\rangle} \\ \text{gave Mary a book} \end{array} \right]\!\right] \left( \left[\!\left[ \begin{array}{c} \mathbf{NP}_{1e} \\ \text{John} \end{array} \right]\!\right] \right)$ (by FA)

2. $= [\lambda z_e.\,\mathsf{T} \text{ iff } z \text{ gave b to m}](\mathrm{j})$ (by b., NB)

3. $= \mathsf{T} \text{ iff j gave b to m}$ (by β-reduction)

So John gave Mary a book is true iff John gave a book to Mary. Di-transitives like *give* also exhibit what is called 'dative alternation'. These are also called 'double object constructions' (see Larson 1988 for instance) given in (4)

(4) a. John gave Mary a book.
b. John gave a book to Mary.




**Your task**
Work *top down*. Fill in the types of higher VPs then,

a. to Mary is a PP. What type must the PP be for **FA** to apply at the higher VP? _____

b. Now what type is to? _____

c. What type is the lower VP? _____

d. What type is gave? _____

**Given**

$[\![\text{John}]\!] = \text{j}$
$[\![\text{Mary}]\!] = \text{m}$
$[\![\text{a book}]\!] = \text{b}$

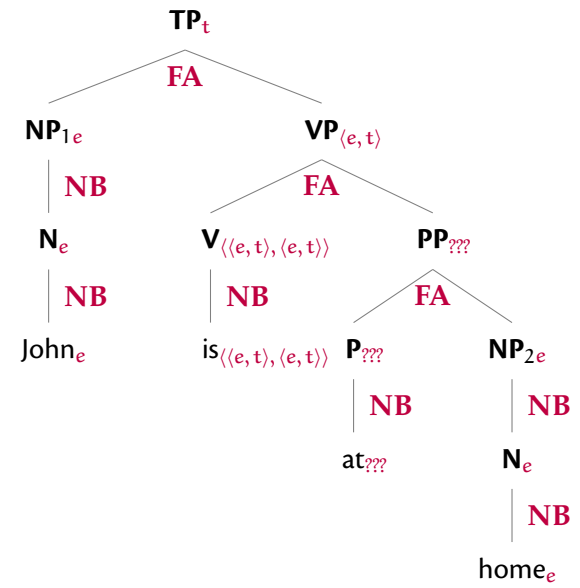
**Write down**

$[\![\text{gave}]\!] =$ _____
$[\![\text{to}]\!] =$ _____

The structure now has a PP hosting the indirect object. Would it affect our computation? What role is *to* playing here? How does it affect its argument?

Now consider the preposition in (5). If we supposed that the entry and type for at is the same as to in (4), what will happen? What do we predict its role to be in the sentence?

(5) John is at home.




**Given**

$[\![\text{John}]\!] = \text{j}$
$[\![\text{home}]\!] = \text{h}$
$[\![\text{is}]\!] = [\lambda f : f \in D_{\langle e,t\rangle} . f]$

**Your task**

$[\![\text{at}]\!] = ??$

a. is is an identity function on predicates. Given its type, what type must PP be? _____

b. What type is at? _____

c. Derive $[\![\text{TP}]\!]$, labelling each step. What are the truth conditions for (5)?

## References

Larson, Richard K. 1988. On the double object construction. *Linguistic Inquiry* 19(3). 335–391. http://www.jstor.org/stable/25164901.