Introduction to Semantic Theory
Non-verbal predicates

Dr. Sarah Zobel

Class: May 25, 2016

Connecting back to the previous lecture – I

Central result: the set-up of the derivational system

- Setting the truth conditions of a sentence as the thing to be derived by the formal system.

  (1) Formal notation: $[\![X]\!]^w = 1$ iff Y

- The introduction of semantic types as a way to represent the logical structure of extensions.
  - Basic types: e (individuals/entities), t (truth values)
  - Functional types:
    for arbitrary types σ, τ, $\langle \sigma, \tau \rangle$ is also a type

Connecting back to the previous lecture – II

- The introduction of the derivational rules (NN) and (FA):

(2) Non-branching nodes (NN):
For a non-branching node α with the single daughter β, $[\![\alpha]\!]^w = [\![\beta]\!]^w$

(3) Function application (FA):
For a branching node α with the set of daughters $\{\beta, \gamma\}$, where β is of type $\langle\sigma, \tau\rangle$ and γ is of type σ, then $[\![\alpha]\!]^w = [\![\beta]\!]^w([\![\gamma]\!]^w)$

- The introduction of λ-conversion:

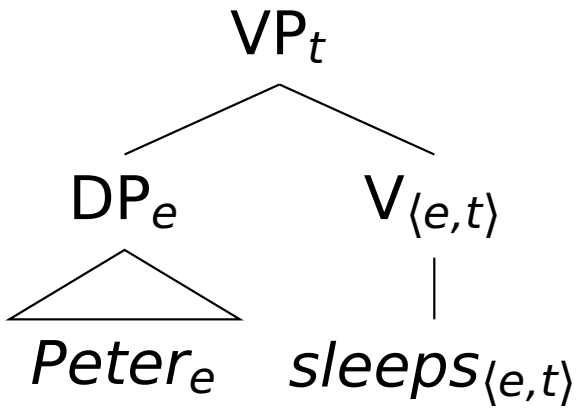
(4) $\left[\lambda x.\ f[x]\right](y) = f[y]$

Aim for today

The aim for today: to explore for which types of sentences, we can derive the truth conditions with the system as it stands now.

- ⇒ We take a look at non-verbal predicates in copular clauses: nouns, adjectives, and PPs.
- ⇒ We will see: with the machinery in place, we can make proposals for the extension and its logical structure of new functional(!) lexical elements for which providing an extension intuitively is hard.

The set-up: *Peter sleeps*



- $[\![\textit{Peter}]\!]^w = \text{Peter}$
- $[\![\textit{sleeps}]\!]^w = \lambda x.\text{sleep'}(x)(w)$

The full derivation: *Peter sleeps*

$$[\![\text{VP}[\text{DP}[\textit{Peter}]\ \text{V}[\textit{sleeps}]]]\!]^w \overset{2\times(NN)}{=\!=} [\![\text{VP}[\textit{Peter}\ \textit{sleeps}]]\!]^w$$

$$\overset{(FA)}{=\!=} [\![\textit{sleeps}]\!]^w([\![\textit{Peter}]\!]^w)$$

$$= [\lambda x.\text{sleep'}(x)(w)](\text{Peter})$$

$$\overset{\lambda}{=\!=} 1 \text{ iff } \text{sleep'}(\text{Peter})(w)$$

The left part and the last line of the right part give the truth conditions of *Peter sleeps* in formal notation!

Extending to transitive verbs – I

Last time we derived the extension for the VP of the sentence '*Peter likes Mary*':

$$\left[\!\left[\begin{array}{c} \text{VP} \\ \text{V} \quad \text{DP} \\ \text{likes} \quad \text{Mary} \end{array}\right]\!\right]^w \overset{2\times(NN)}{=} \left[\!\left[\begin{array}{c} \text{VP} \\ \text{likes} \quad \text{Mary} \end{array}\right]\!\right]^w$$

$$\overset{(FA)}{=} [\![\textit{likes}]\!]^w([\![\textit{Mary}]\!]^w)$$

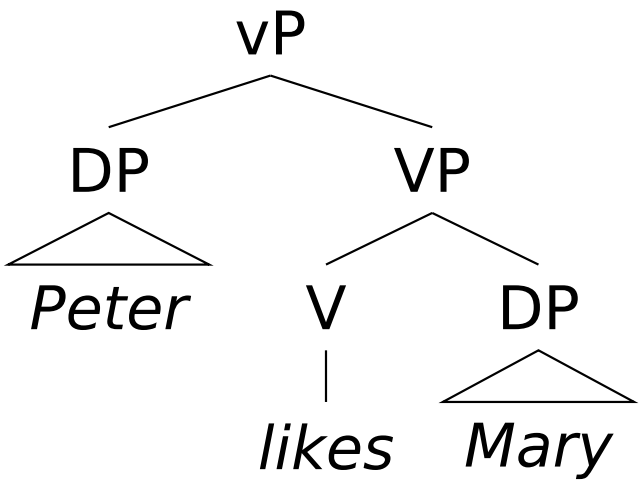
$$= [\lambda y.\lambda x.\text{like'}(y)(x)(w)](\text{Mary})$$

$$\overset{\lambda}{=} [\lambda x.\text{like'}(\text{Mary})(x)(w)]$$

Extending to transitive verbs – II

We are still missing one step to derive the truth conditions for '*Peter likes Mary*'.

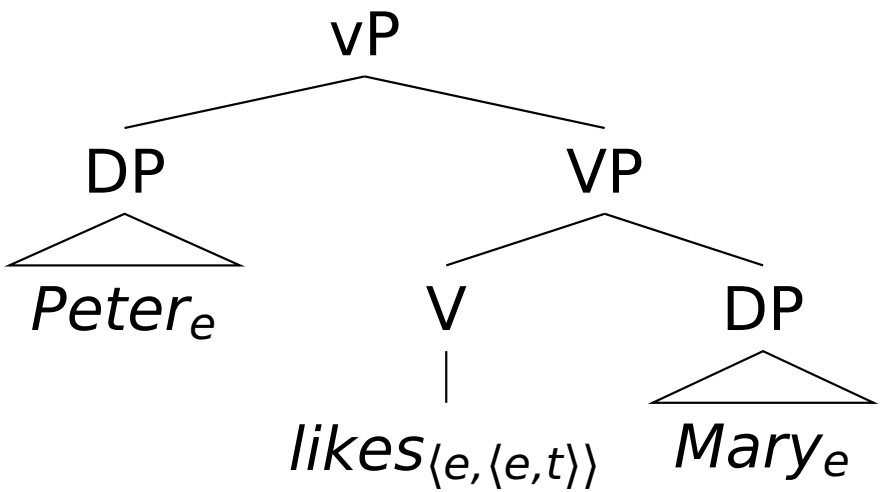
⇒ Determine the types for the full structure:



Extending to transitive verbs – II

We are still missing one step to derive the truth conditions for '*Peter likes Mary*'.

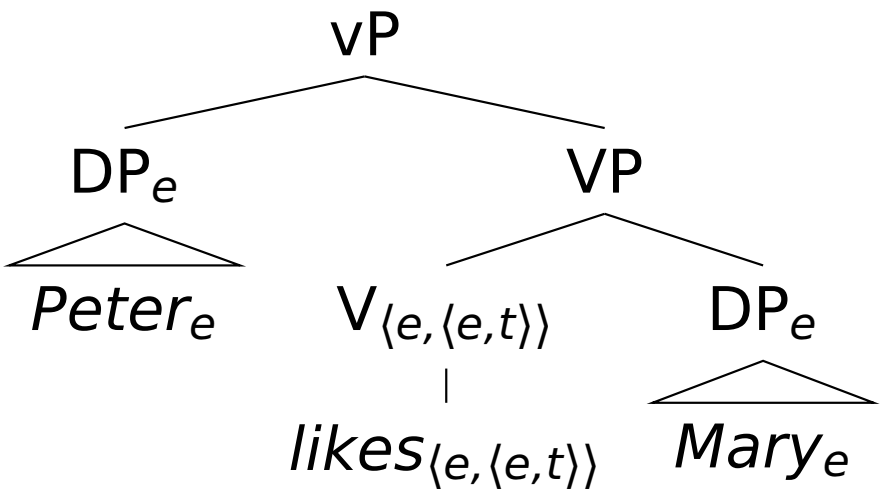
⇒ Determine the types for the full structure:



Extending to transitive verbs – II

We are still missing one step to derive the truth conditions for '*Peter likes Mary*'.

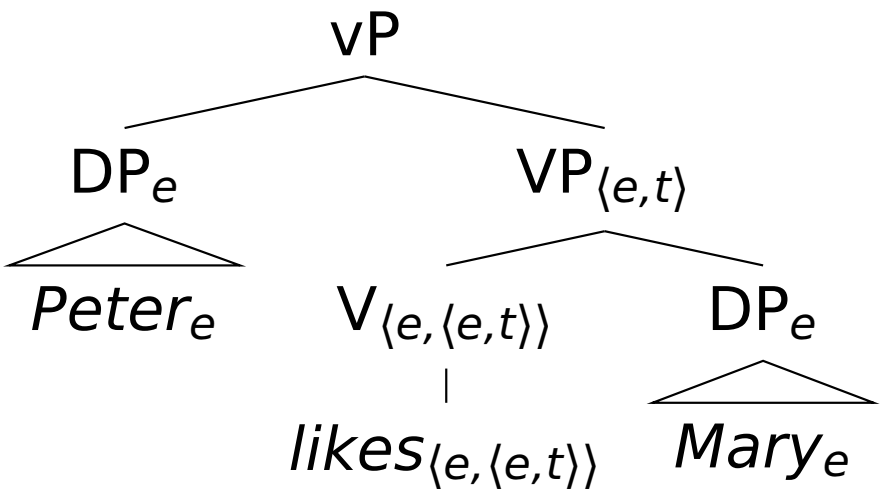
⇒ Determine the types for the full structure:



Extending to transitive verbs – II

We are still missing one step to derive the truth conditions for '*Peter likes Mary*'.

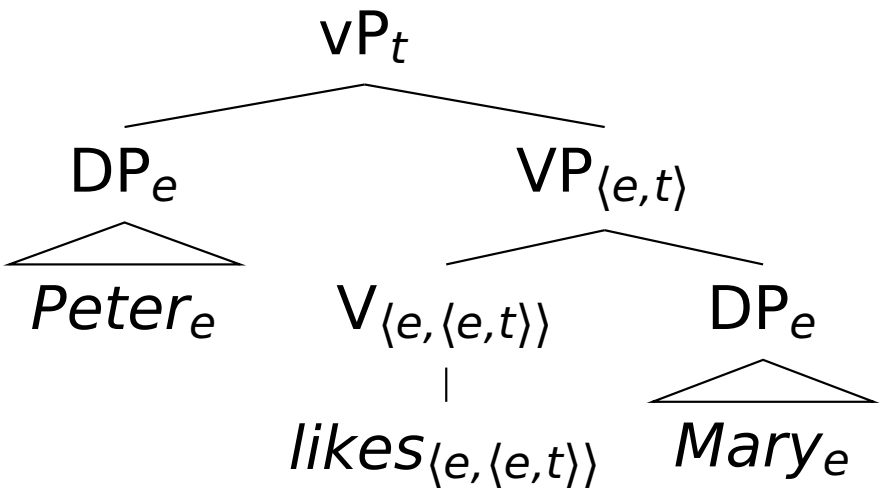
⇒ Determine the types for the full structure:



Extending to transitive verbs – II

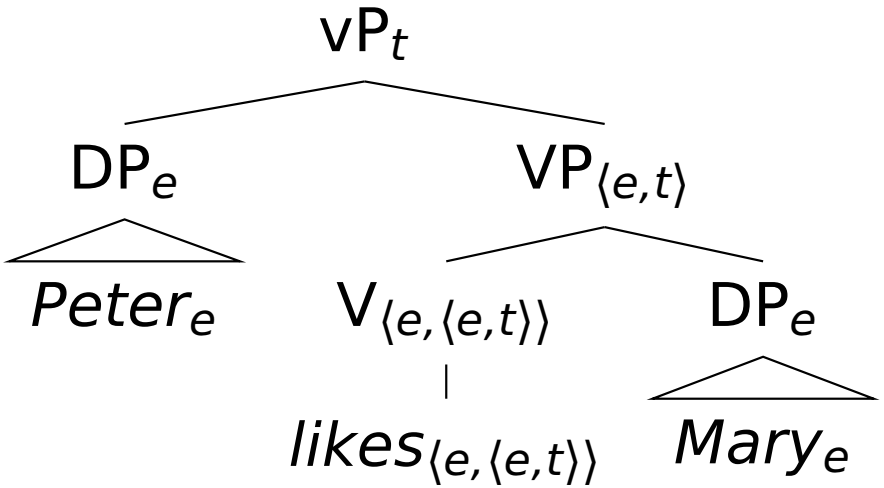
We are still missing one step to derive the truth conditions for '*Peter likes Mary*'.

⇒ Determine the types for the full structure:



Extending to transitive verbs – III

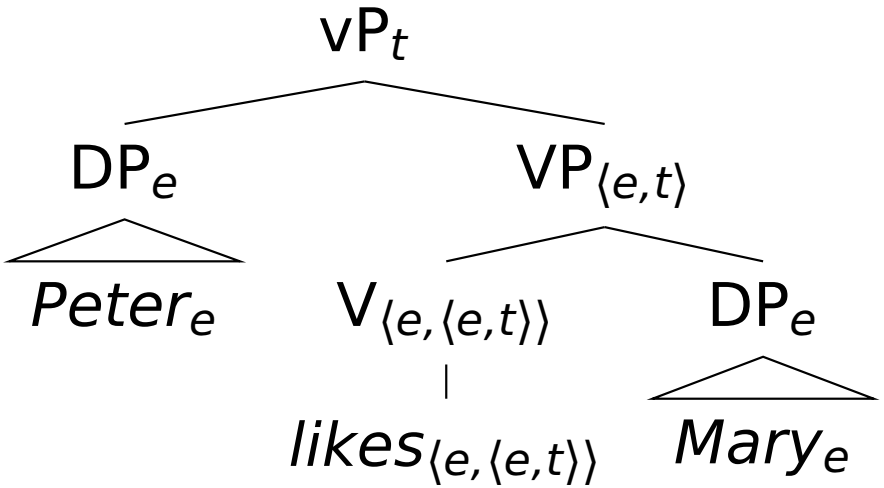
We determined the extension of the subtree headed by VP by applying (NN), (FA), and performing λ-conversion: $[\![\text{VP}]\!]^w = \lambda x.\text{like'}(\text{Mary})(x)(w)$



To derive the truth conditions for the full sentence, we need to apply (NN) and (FA) once more.
The input for the second application of (FA) will be: $[\![\text{DP}]\!]^w \overset{(NN)}{=} [\![\textit{Peter}]\!]^w$ and $[\![\text{VP}]\!]^w$.

Extending to transitive verbs – III

We determined the extension of the subtree headed by VP by applying (NN), (FA), and performing λ-conversion:
$[\![\text{VP}]\!]^w = \lambda x.\text{like'}(\text{Mary})(x)(w)$



To derive the truth conditions for the full sentence, we need to apply (NN) and (FA) once more.
The input for the second application of (FA) will be: $[\![\text{DP}]\!]^w \overset{(NN)}{=} [\![\textit{Peter}]\!]^w$ and $[\![\text{VP}]\!]^w$.

$\Rightarrow$ (NN) and (FA) are recursive rules

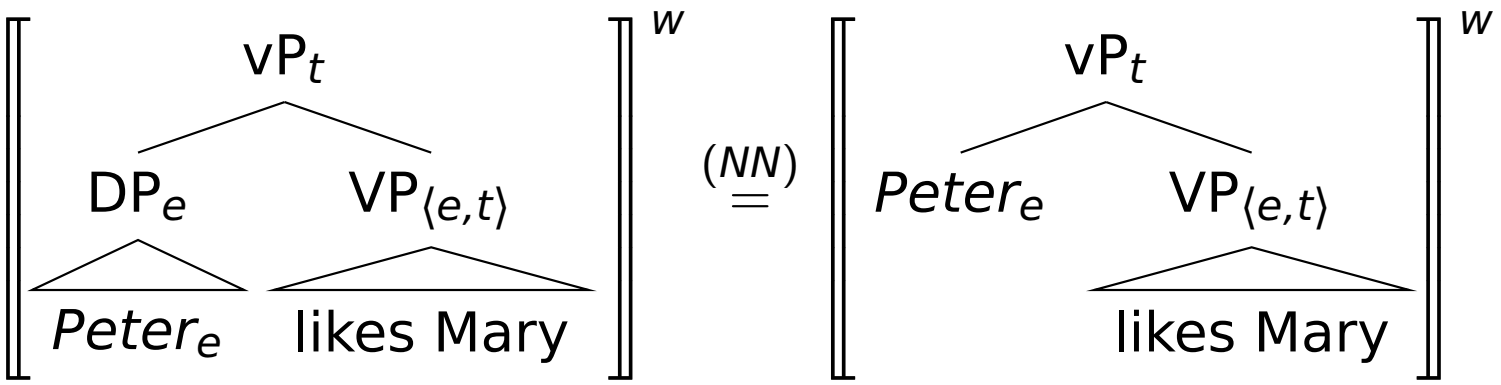
Recursive rules

All derivational rules that we will formulate are recursive.

(5) A rule is recursive iff it can take its own output as input.

In the case of a sentence containing a transitive verb, we saw the recursivity of (FA): the second application of (FA) to derive the truth conditions of the sentence used the output of the application of (FA) (= the extension of the VP) as one of its inputs.

The rest of the derivation: *Peter likes Mary*



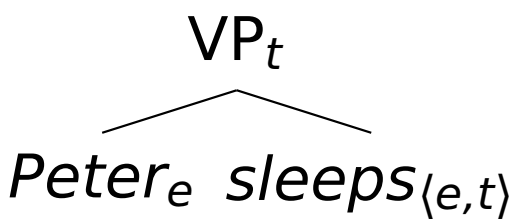
$$\overset{(FA)}{=} [\![\text{VP}]\!]^w([\![\textit{Peter}]\!]^w)$$

$$= [\lambda x.\text{like'}(\text{Mary})(x)(w)](\text{Peter})$$

$$\overset{\lambda}{=} 1 \text{ iff like'}(\text{Mary})(\text{Peter})(w)$$

Non-verbal predicates – I

Previous observation: the extensions of certain nouns and adjectives have the same logical structure as the extensions of intransitive verb (= they have the same type).



Since (FA) only requires the types of two expressions to fit to be applied – one can take the other as an argument – we would expect to see (ungrammatical) combinations like:

(6) **Peter tall*, **Peter man*

Non-verbal predicates – II

This might seem like a problem for our system.

What is the composition of *Peter* and *tall* OR *Peter* and *man* proposed to be in our system?

Non-verbal predicates – II

This might seem like a problem for our system.

What is the composition of *Peter* and *tall* OR *Peter* and *man* proposed to be in our system?

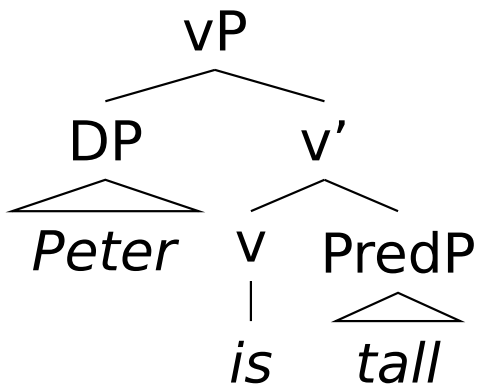
However, there is a type of sentence for which this prediction of our system is indeed positive: copular sentences.

(7) a. *Peter is tall.*
b. *Peter is a man.*

In copular sentences the adjective and the noun – together with the copula *be* – form the "verbal part" of the sentence.

LF of an adjectival copular sentence

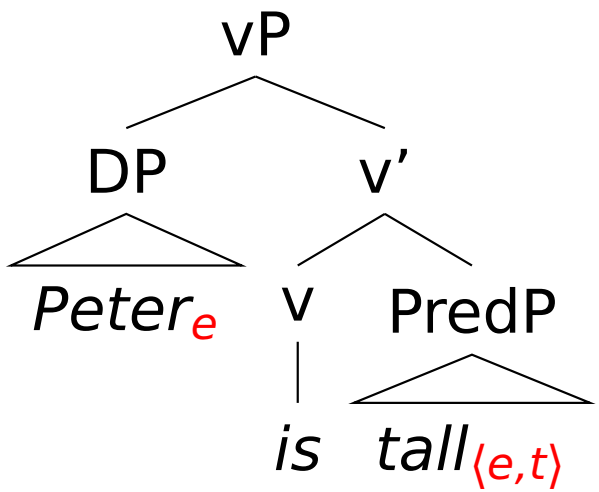
For the structure below, we already have hypotheses about the extensions and types for *Peter* and *tall*.



Is there a way to determine from the structure what the logical structure of the copula needs to be so that we can use (NN) and (FA) to derive the truth conditions of this sentence?

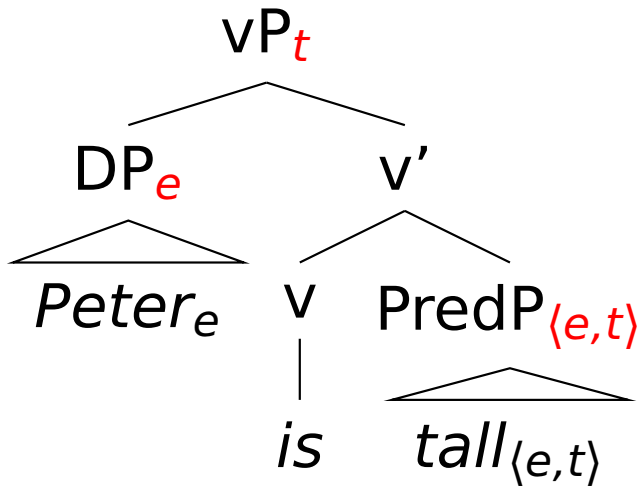
Deriving the type of *is*

We can use the type conditions of (NN) – which makes no restrictions – and of (FA) – which requires a function-argument-configuration to derive the type for the copula that we would need for (NN) and (FA) to be applicable.



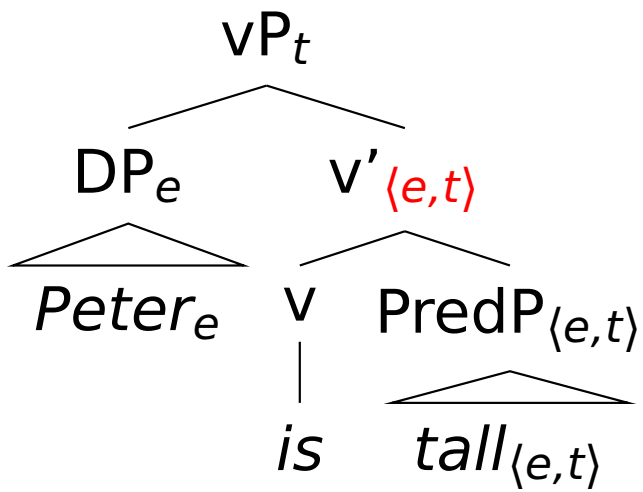
Deriving the type of *is*

We can use the type conditions of (NN) – which makes no restrictions – and of (FA) – which requires a function-argument-configuration to derive the type for the copula that we would need for (NN) and (FA) to be applicable.



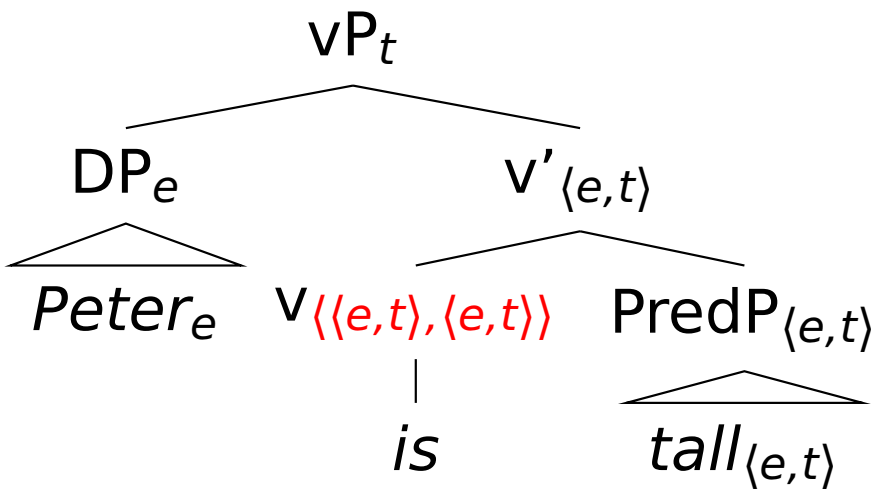
Deriving the type of *is*

We can use the type conditions of (NN) – which makes no restrictions – and of (FA) – which requires a function-argument-configuration to derive the type for the copula that we would need for (NN) and (FA) to be applicable.



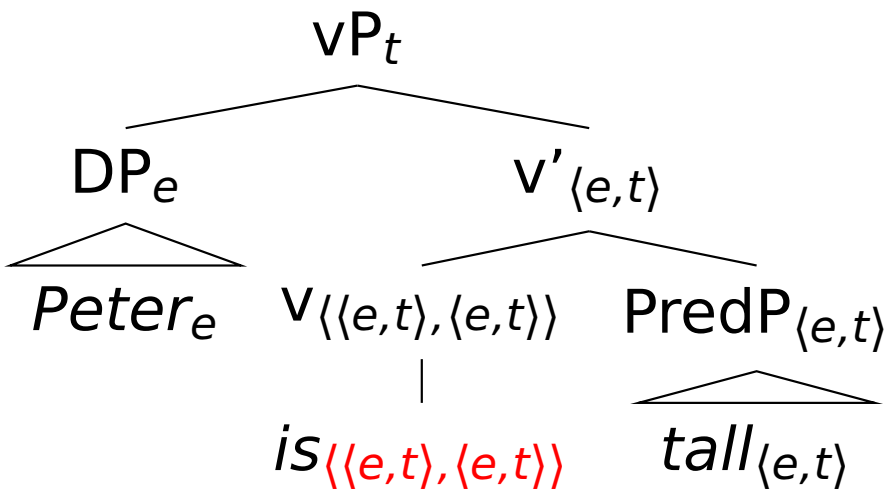
Deriving the type of *is*

We can use the type conditions of (NN) – which makes no restrictions – and of (FA) – which requires a function-argument-configuration to derive the type for the copula that we would need for (NN) and (FA) to be applicable.



Deriving the type of *is*

We can use the type conditions of (NN) – which makes no restrictions – and of (FA) – which requires a function-argument-configuration to derive the type for the copula that we would need for (NN) and (FA) to be applicable.



The contribution of *is* – I

Now we have a hypothesis about what is the type of *is* – the logical structure of the extension of *is*.

But what is the extension of *is*? What does *is* contribute to the truth conditions of the sentence?

The contribution of *is* – II

Since the adjective already contributes a function that can take the given individual as an argument – AND the result would be the condition that this individual be in the set of individuals given by the adjective – the copula *is* does not seem to contribute anything of substance.

(8) $[\![is]\!]^w = \lambda P.\ P$ where P is of type $\langle e, t \rangle$

⇒ the extension of the copula is an identity function over functions of type $\langle e, t \rangle$.

The contribution of *is* – II

Since the adjective already contributes a function that can take the given individual as an argument – AND the result would be the condition that this individual be in the set of individuals given by the adjective – the copula *is* does not seem to contribute anything of substance.

(8) $[\![is]\!]^w = \lambda P.\ P$ where P is of type $\langle e, t \rangle$

⇒ the extension of the copula is an identity function over functions of type $\langle e, t \rangle$.

Does the extension proposed for *is* match our hypothesis regarding its type?

Nominal copular sentences

We have seen that we also find nominal copular sentences in addition to adjectival copular sentences.

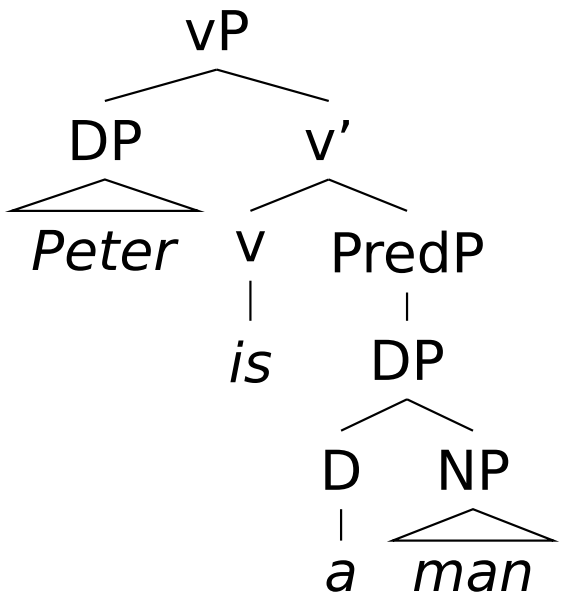
(9) *Peter is a man.*

After having determined the semantic type and extension for the copula *is*, we have hypotheses for three out of four lexical items for the sentence above.

But: We are still missing the type and extension for the determiner *a*.

LF of the nominal copular clause

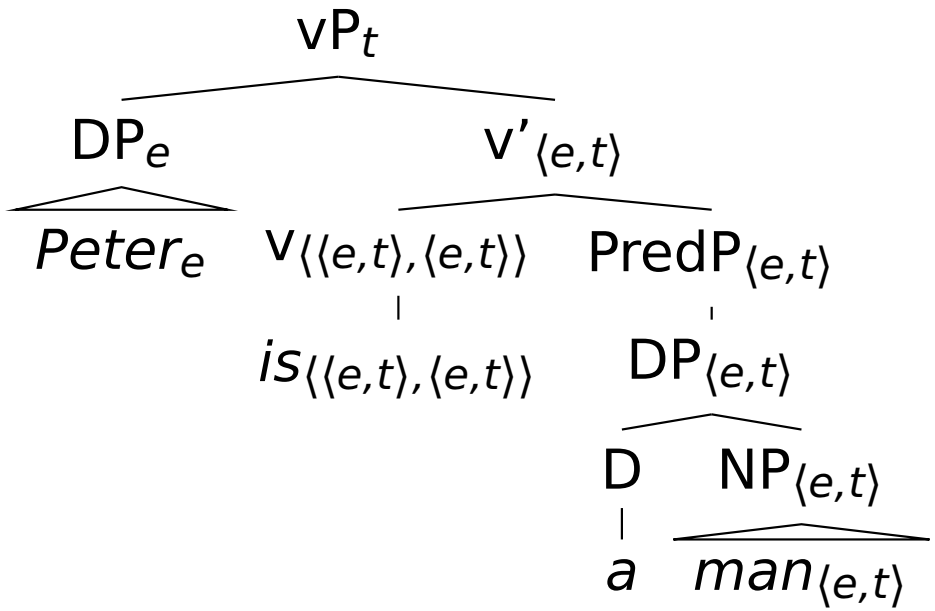
Consider the following structure and what we have said about the types of the parts we have already seen.



What could be the type of the determiner *a*?

The type of *a* – I

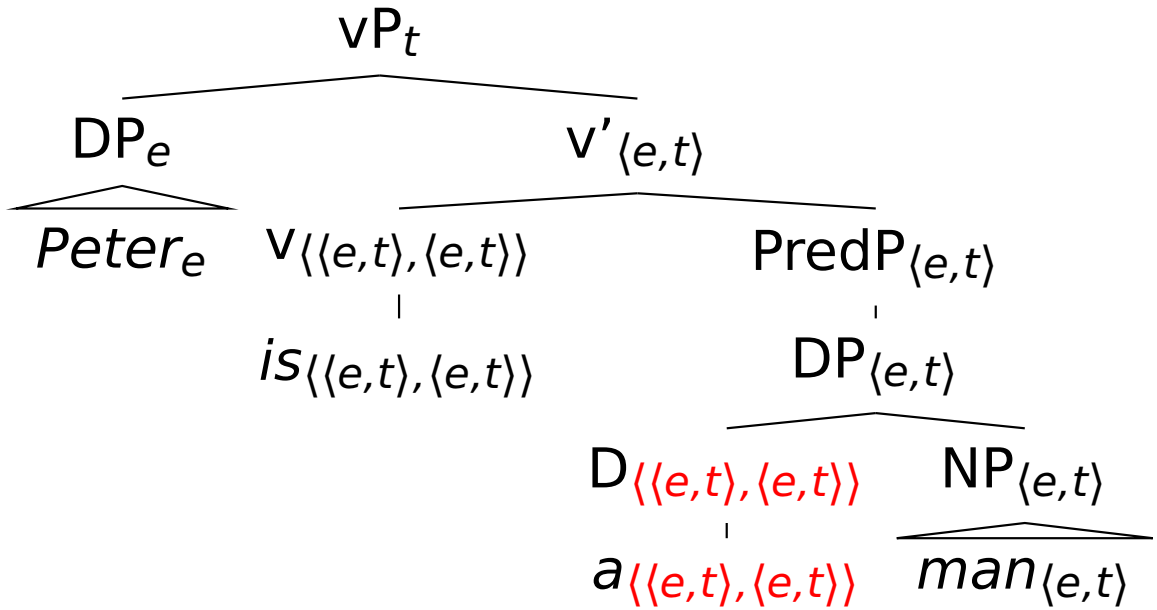
If we fill in all of the types which we already know or can derive:



What does the type of the determiner *a* have to be if we want to apply (FA) for the branching node under DP?

The type of *a* – II

The type of *a* matches that of *is*:



What could be the extension of the determiner *a*?

The contribution of *a*

Since the noun already contributes a function that can take the given individual as an argument – AND the result would be the condition that this individual be in the set of individuals given by the noun – the determiner *a*, like the copula *is*, does not seem to make any notable contribution.

(10) $[\![a]\!]^w = \lambda P.\ P$ where P is of type $\langle e, t \rangle$

⇒ the extension of the determiner *a* in a nominal copular sentence is an identity function over functions of type $\langle e, t \rangle$.

The contribution of *a*

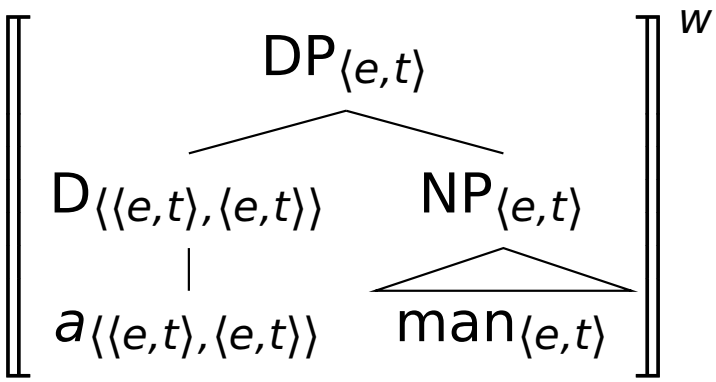
Since the noun already contributes a function that can take the given individual as an argument – AND the result would be the condition that this individual be in the set of individuals given by the noun – the determiner *a*, like the copula *is*, does not seem to make any notable contribution.

(10) $[\![a]\!]^w = \lambda P.\ P$ where P is of type $\langle e, t \rangle$

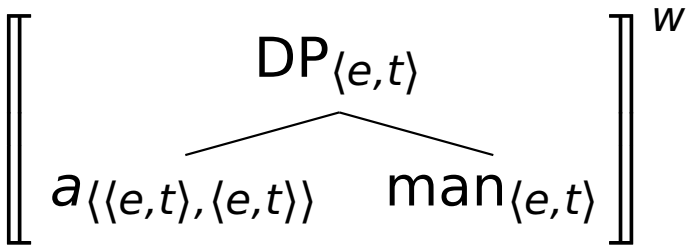
⇒ the extension of the determiner *a* in a nominal copular sentence is an identity function over functions of type $\langle e, t \rangle$.

Does the extension proposed for *a* match our hypothesis regarding its type?

Deriving the extension of '*a man*'



$\overset{2\times(NN)}{=}$



$$\overset{(FA)}{=} [\![a]\!]^w([\![man]\!]^w)$$

$$= [\lambda P.\ P]([\lambda x.\ \text{man'}(x)(w)])$$

$$\overset{\lambda}{=} \lambda x.\ \text{man'}(x)(w)$$

Teaser: the problem of attributive adjectives

We have hypotheses for the types and extensions of nouns and adjectives: we should be able to derive the extension of nouns modified by attributive adjectives.

(11) a. *gray cat*
b. *small child*

⇒ However: There seems to be a problem with combining the extensions of nouns with those of adjectives, given the rules (NN) and (FA). Exercise 3 b) of problem set 5!

⇒ Solution: we need to either change our hypotheses about adjectives or to introduce a new derivation rule.

Summary

- The rules (NN) and (FA) are recursive, i.e., they can take their outputs of previous applications as inputs for a new application of the rule. All derivation rules that we will meet are recursive.
- To treat sentences containing transitive verbs, we do not need any new assumptions; only recursive application of (NN) and (FA).
- To deal with non-verbal predicates in copular sentences, we used the formal system to derive hypotheses regarding the semantic types of the copula *is* and the determiner *a*.
- Following (more or less) intuitive considerations, we also proposed extensions for these two lexical elements.