Introduction to Formal Semantics: RECAP

Volha Petukhova & Nicolaie Dominik Dascalu

Spoken Language Systems Group
Saarland Univeristy

11.07.2022

Overview for today

- All 10 topics: summary of important issues
- Question session
- Some formalities

Topic 1: Meaning & Form

Meaning representation

- Structures from set of symbols
    - representational vocabulary
- Symbol structures correspond to:
    - objects
    - properties of objects
    - relations among objects
- Can be viewed as:
    - representation of meaning of linguistic input
    - representation of state of world

Topic 1: Meaning & Form (models)

A model is an abstract mathematical structure that we construct for describing hypothetical situations. Models are used for analyzing natural language expressions (words, phrases and sentences) by associating them with abstract objects.

A semantic theory T satisfies the **truth-conditional criterion** (TCC) for sentences S_1 and S_2 if the following two conditions are equivalent:

1. Sentence S_1 intuitively entails S_2.
2. For all models M in T: $[\![S_1]\!]^M \leq [\![S_2]\!]^M$

Topic 1: Meaning & Form

Models: truth values

(1) Tina is tall and thin.

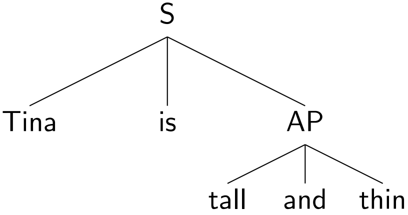
(2) Tina is tall.

(1) $\Longrightarrow$ (2)

Expression	Cat.	Type	Abstract denotation	Denotations in example models with $E = \{a, b, c, d\}$		
				M_1	M_2	M_3
Tina	PN	entity	**tina**	a	b	b
tall	A	set of entities	**tall**	{b,c}	{b,d}	{a,b,d}
thin	A	set of entities	**thin**	{a,b,c}	{b,c}	{a,c,d}
tall and thin	AP	set of entities	AND(**tall**, **thin**)	{b,c}	{b}	{a,d}
Tina is thin	S	truth-value	IS(**tina**, **thin**)	1	1	0
Tina is tall and thin	S	truth-value	IS(**tina**, AND(**tall**, **thin**))	0	1	0

Topic 1: Meaning & Form (compositionality)

a.

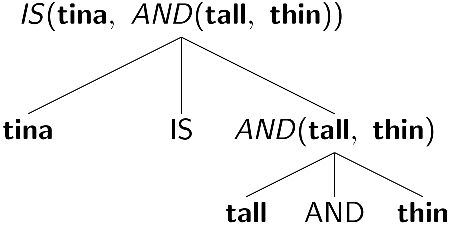


S

Tina is AP

tall and thin

b.



IS(**tina**, *AND*(**tall**, **thin**))

tina IS *AND*(**tall**, **thin**)

tall AND **thin**

Compositionality: the denotation of a complex expressions is determined by the denotations of its immediate parts and the ways they combine with each other

Topic 2: Inference

p entails q

- whenever p is true, q must be true
- a situation describable by p must be also describable by q
- p and $\neg q$ is contradictory (cannot be true in any situation)

Entailment: (i) indefeasible; (ii) speakers intuitively accept S2 whenever they accept S1.

Presupposition a special type of entailment

- p entails q iff q is true whenever p is true
- p presupposes q iff q is true whenever p or $\neg p$ is true

Presupposition projection (tests). Presupposition are defeasible (conntextual and structural defeasibility; examples).

Presupposition (cont.)

Presuppositions are special entailments. Ordinary entailments disappear when the sentence is *negated*, turned into a *question*, or embedded under a *possibility modal* like maybe:

Example

Presupposition (cont.)

Presuppositions are special entailments. Ordinary entailments disappear when the sentence is *negated*, turned into a *question*, or embedded under a *possibility modal* like maybe:

Example

1. Alice has a cute dog.

Presupposition (cont.)

Presuppositions are special entailments. Ordinary entailments disappear when the sentence is *negated*, turned into a *question*, or embedded under a *possibility modal* like maybe:

Example

1. Alice has a cute dog.
2. $\Longrightarrow$ Alice has a dog.

Presupposition (cont.)

Presuppositions are special entailments. Ordinary entailments disappear when the sentence is *negated*, turned into a *question*, or embedded under a *possibility modal* like maybe:

Example

1. Alice has a cute dog.
2. $\Longrightarrow$ Alice has a dog.
3. b. Alice doesn't have a cute dog.

Presupposition (cont.)

Presuppositions are special entailments. Ordinary entailments disappear when the sentence is *negated*, turned into a *question*, or embedded under a *possibility modal* like maybe:

Example

1. Alice has a cute dog.
2. $\Longrightarrow$ Alice has a dog.
3. b. Alice doesn't have a cute dog.
4. c. Does Alice have a cute dog?

Presupposition (cont.)

Presuppositions are special entailments. Ordinary entailments disappear when the sentence is *negated*, turned into a *question*, or embedded under a *possibility modal* like maybe:

Example

1. Alice has a cute dog.
2. $\Longrightarrow$ Alice has a dog.
3. b. Alice doesn't have a cute dog.
4. c. Does Alice have a cute dog?
5. d. Maybe Alice has a cute dog.

Presupposition (cont.)

Presuppositions are special entailments. Ordinary entailments disappear when the sentence is *negated*, turned into a *question*, or embedded under a *possibility modal* like maybe:

Example

1. Alice has a cute dog.
2. $\Longrightarrow$ Alice has a dog.
3. b. Alice doesn't have a cute dog.
4. c. Does Alice have a cute dog?
5. d. Maybe Alice has a cute dog.
6. (b–d): $\not\Longrightarrow$ Alice has a dog.

Presupposition (cont.)

But presuppositions survive in all of these environments (we also say: presuppositions project):

Example

Presupposition (cont.)

But presuppositions survive in all of these environments (we also say: presuppositions project):

Example

1. Bob's sister is a professional wrestler.

Presupposition (cont.)

But presuppositions survive in all of these environments (we also say: presuppositions project):

Example

1. Bob's sister is a professional wrestler.
2. $\Longrightarrow$ Bob has a sister.

Presupposition (cont.)

But presuppositions survive in all of these environments (we also say: presuppositions project):

Example

1. Bob's sister is a professional wrestler.
2. $\Longrightarrow$ Bob has a sister.
3. b. Bob's sister is not a professional wrestler.

Presupposition (cont.)

But presuppositions survive in all of these environments (we also say: presuppositions project):

Example

1. Bob's sister is a professional wrestler.
2. $\Longrightarrow$ Bob has a sister.
3. b. Bob's sister is not a professional wrestler.
4. c. Is Bob's sister a professional wrestler?

Presupposition (cont.)

But presuppositions survive in all of these environments (we also say: presuppositions project):

Example

1. Bob's sister is a professional wrestler.
2. $\Longrightarrow$ Bob has a sister.
3. b. Bob's sister is not a professional wrestler.
4. c. Is Bob's sister a professional wrestler?
5. d. Maybe Bob's sister is a professional wrestler.

Presupposition (cont.)

But presuppositions survive in all of these environments (we also say: presuppositions project):

Example

1. Bob's sister is a professional wrestler.
2. $\Longrightarrow$ Bob has a sister.
3. b. Bob's sister is not a professional wrestler.
4. c. Is Bob's sister a professional wrestler?
5. d. Maybe Bob's sister is a professional wrestler.
6. (b–d): $\Longrightarrow$ Bob has a sister.

Topic 2: Inference (Semantic Presupposition: problems)

Problem 1 Presupposition failure (= the presupposition is false in context)

(1) King of France is bald.
When uttered on May 13 2005, the presupposition is false

Problem 2 Presupposition cancellation (= the presupposition is “removed” in context)

(2) I don’t know that Bill came.
This utterance does not presuppose that speaker knows that Bill came.

Topic 2: Inference (Russell-Strawson debate)

(3) The king of France is bald.

Russell:

- is quantificational. It is logically equivalent to: “exactly one entity has the property King of France, and that entity is bald”
- Thus, if there is no unique King of France, (3) is **false**.
- $\exists x.KOF = \{x\} \ \wedge \ \mathbf{bald}(x)$

Strawson:

- Any use of (3) raises the following presupposition: “exactly one entity, call it x, has the property King of France”
- Under this presupposition, (3) means: “x is bald”
- Thus, if there is no unique King of France, (3) is neither **true** nor **false**.
- $\exists x.KOF = \{x\} \ : \exists xKOF = \{x\} \ \wedge \ \mathbf{bald}(x)$

Topic 2: Inference (Trivalent Strawsonian semantics)

$[\![S]\!]^M = \phi : \psi$

ϕ indicates whether S is assertible in M

ψ indicates whether S is true in M

Definition: the colon operator ('transplication'):

$$(\phi : \psi) = \begin{cases} \psi & \phi = 1 \\ * & \phi = 0 \end{cases}$$

where ϕ and ψ are bivalent truth-values.

Blamey (1986) suggests the possibility of reading the transplication connective as a type of conditionals of the form 'if ϕ then ψ' which are neither true nor false when ϕ is false. They are also neither true nor false when either ϕ or ψ is neither true nor false.

Topic 3: Predication & Argumentation

Anatomy of First-Order-Logic (syntax and semantics): variables, constants, predicates, functions, logical connectives, quantifiers

Predicate Logics: well-formed formulas, scope and binding

Able to translate to FOL and to NL

Topic 4: Typed Lambda (abstraction from fully specified FOL)

Example

John loves Mary

$Loves(j, m)$

$Loves(\_\_, m)$

to abstract OVER the missing piece, ABSTRACTION OPERATOR λ is used

Example

John loves Mary

$Loves(j, m)$

$Loves(\_\_, m)$

$\lambda x.Loves(x, m)$

This expression denotes a function from an individual to truth-value

Topic 4: Typed Lambda (Lambda Abstraction: Types)

Syntactic categories of languages L_{Pred} are terms, predicates and formulas

Language L_λ has a set of TYPES which are recursively specified (of arbitrary complexity and depth), with two BASIC TYPES:

- *e* **e**ntities for individuals corresponding to terms in L_{Pred}
- *t* **t**ruth values for formulas

and FUNCTION TYPES: $< e, t >$ denoting functions from individuals to truth values.
A set of types is defined recursively:

- *e* is a type
- *t* is a type
- if σ is a type and τ is a type then $< \sigma, \tau >$ is a type
- nothing else is a type

Topic 4: Typed Lambda (nested) Lambda Reduction

Example

$\lambda x.\lambda y.Near(x, y)$

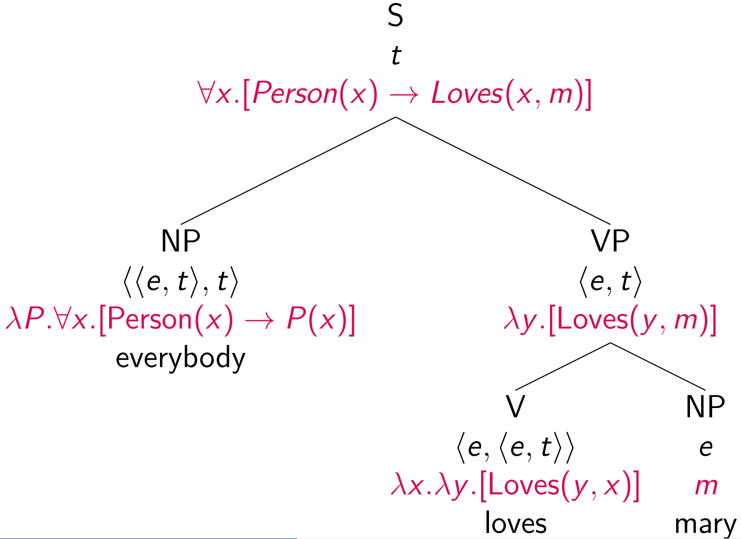
$\lambda x.\lambda y.Near(x, y)(midway)$

$\lambda y.Near(midway, y)$

$\lambda y.Near(midway, y)(chicago)$

$Near(midway, chicago)$

Topic 4: Typed Lambda (trees) Lambda Reduction



S

t

$\forall x.[Person(x) \rightarrow Loves(x, m)]$

NP

$\langle\langle e, t\rangle, t\rangle$

$\lambda P.\forall x.[\text{Person}(x) \rightarrow P(x)]$

everybody

VP

$\langle e, t\rangle$

$\lambda y.[\text{Loves}(y, m)]$

V

$\langle e, \langle e, t\rangle\rangle$

$\lambda x.\lambda y.[\text{Loves}(y, x)]$

loves

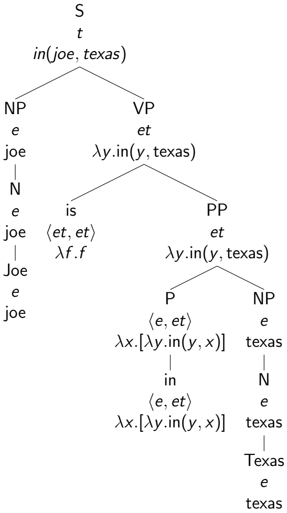
NP

e

m

mary

Topic 4: Typed Lambda (trees)



S
t
$in(joe, texas)$

NP
e
joe

N
e
joe

Joe
e
joe

VP
et
$\lambda y.\text{in}(y, \text{texas})$

is
$\langle et, et \rangle$
$\lambda f.f$

PP
et
$\lambda y.\text{in}(y, \text{texas})$

P
$\langle e, et \rangle$
$\lambda x.[\lambda y.\text{in}(y, x)]$

in
$\langle e, et \rangle$
$\lambda x.[\lambda y.\text{in}(y, x)]$

NP
e
texas

N
e
texas

Texas
e
texas

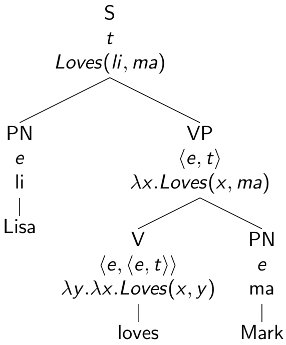
Topic 5: Function Application (compositionality)

Composition Rule 1: Function Application

Let γ be a syntax tree whose sub-trees are α and β where:

- $\alpha \rightsquigarrow \alpha'$ where α' has type $\langle\sigma, \tau\rangle$
- $\beta \rightsquigarrow \beta'$ where β' has type $\langle\sigma\rangle$

then $$\gamma \rightsquigarrow \alpha'(\beta')$$



S
t
$Loves(li, ma)$

PN
e
li
Lisa

VP
$\langle e, t\rangle$
$\lambda x.Loves(x, ma)$

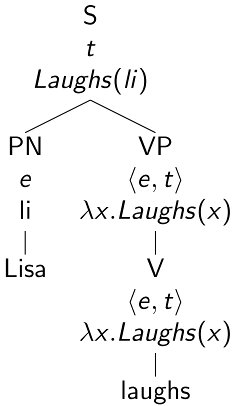
V
$\langle e, \langle e, t\rangle\rangle$
$\lambda y.\lambda x.Loves(x, y)$
loves

PN
e
ma
Mark

Topic 5: Function Application (compositionality)

Composition Rule 2: Non-branching Nodes

If β is a tree whose only daughter is α, where $\alpha \rightsquigarrow \alpha'$ then $\beta \rightsquigarrow \alpha'$



S

t

$Laughs(li)$

PN

e

li

Lisa

VP

$\langle e, t\rangle$

$\lambda x.Laughs(x)$

V

$\langle e, t\rangle$

$\lambda x.Laughs(x)$

laughs

Topic 5: Function Application (compositionality)

- transitive verbs
- identity function
- negation
- indefinite article
- prepositional phrases
- quantification
- generalised quantifiers

Topic 6: Beyond Function Application

Solutions for Adjectives:

(i) generate two translations: one of $\langle e, t\rangle$ for predicative positions and $\langle\langle e, t\rangle, \langle e, t\rangle\rangle$ for attributive positions;

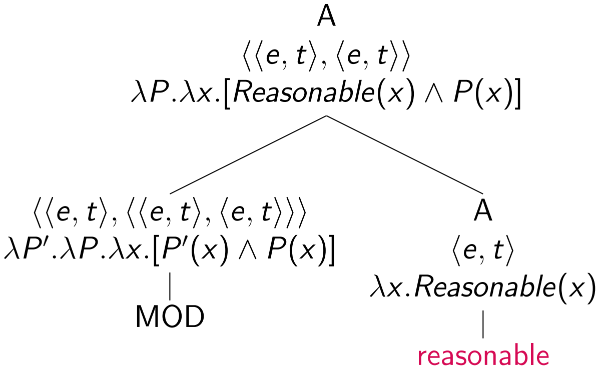
(ii) introduce a new composition rule and give intersective adjectives one single translation

for (i) take one translation to be basic and derive the other one from it with the help of either

a TYPE-SHIFTING RULE (invisible to the syntactic component of the grammar) or

a SILENT OPERATOR (a reflection in the syntax)

Topic 6: Beyond Function Application (Silent Operator)



A

$\langle\langle e,t\rangle,\langle e,t\rangle\rangle$

$\lambda P.\lambda x.[Reasonable(x) \wedge P(x)]$

$\langle\langle e,t\rangle,\langle\langle e,t\rangle,\langle e,t\rangle\rangle\rangle$

$\lambda P'.\lambda P.\lambda x.[P'(x) \wedge P(x)]$

MOD

A

$\langle e,t\rangle$

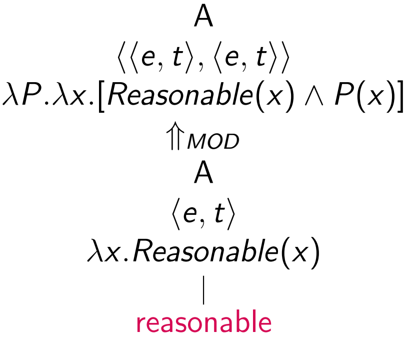
$\lambda x.Reasonable(x)$

reasonable

Topic 6: Beyond Function Application (Type Shifting)

Type-Shifting Rule 1: Predicate-to-modifier shift

If $\alpha \rightsquigarrow \alpha'$, where α' is of type $\langle e, t\rangle$,
then $\alpha \rightsquigarrow \lambda P.[\alpha'(x) \wedge P(x)]$ (as long as P and x are not free in α; in that case, use different variables of the same type).



A

$\langle\langle e, t\rangle, \langle e, t\rangle\rangle$

$\lambda P.\lambda x.[Reasonable(x) \wedge P(x)]$

$\Uparrow_{MOD}$

A

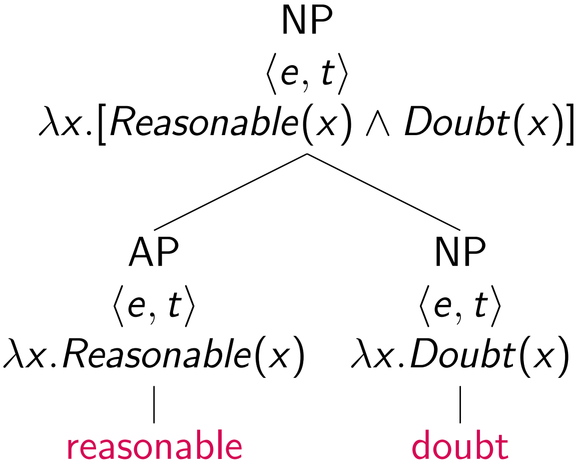
$\langle e, t\rangle$

$\lambda x.Reasonable(x)$

reasonable

Topic 6: Beyond Function Application (Predicate Modification)

(ii) assumption is that all intersective adjectives have translations of a single type, and we eliminate type mismatches via a new composition rule



NP

$\langle e, t \rangle$

$\lambda x.[Reasonable(x) \wedge Doubt(x)]$

AP

$\langle e, t \rangle$

$\lambda x.Reasonable(x)$

reasonable

NP

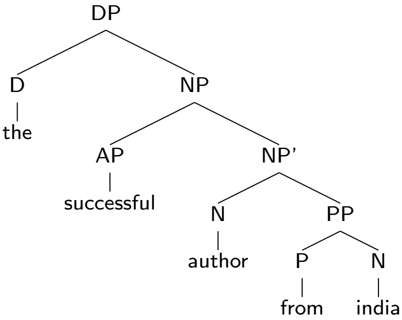
$\langle e, t \rangle$

$\lambda x.Doubt(x)$

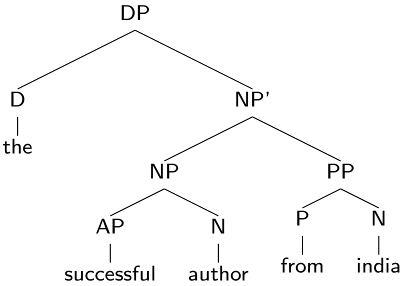
doubt

Florian vs Sandra: *the successful author from India*

a.

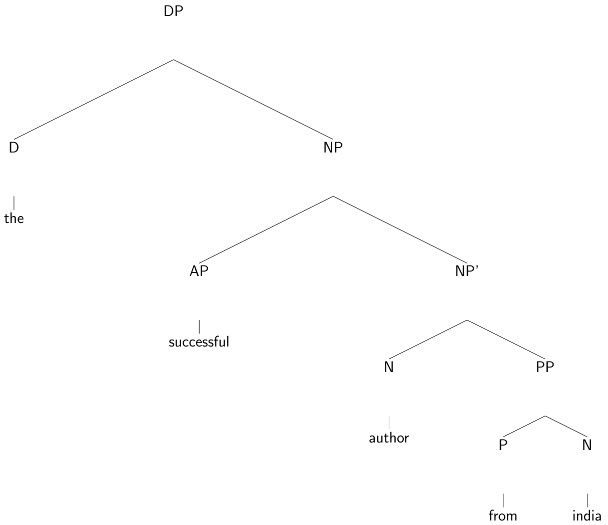


b.



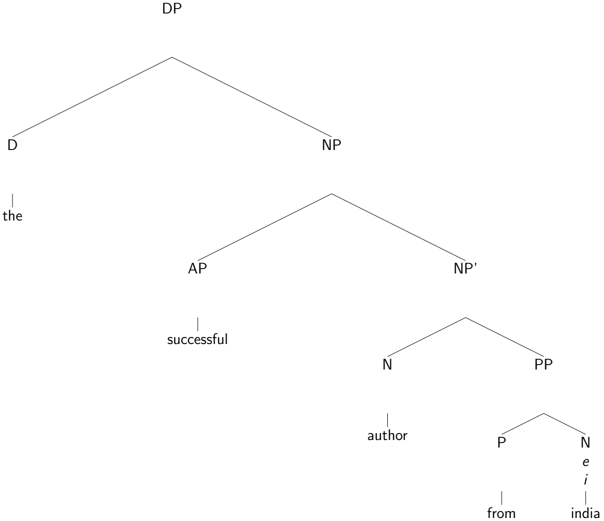
Florian vs Sandra: *the successful author from India*

a.

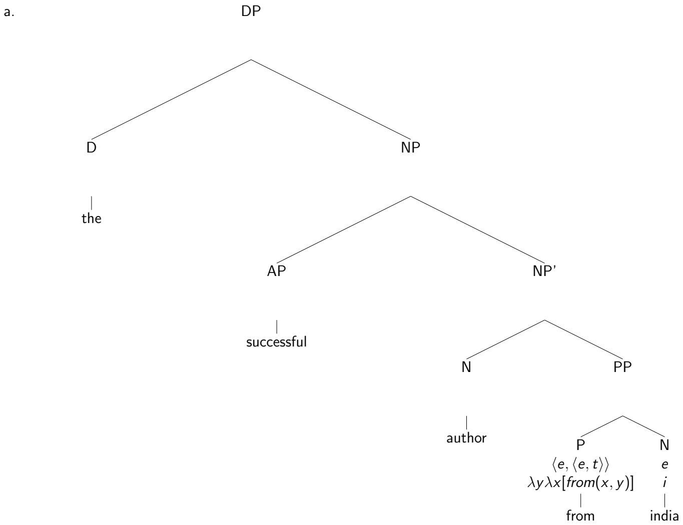


Florian vs Sandra: *the successful author from India*

a.

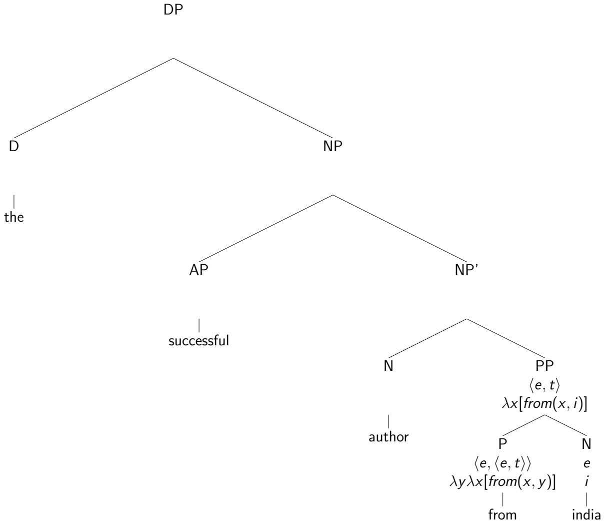


Florian vs Sandra: *the successful author from India*


a.
DP
D
NP
the
AP
NP'
successful
N
PP
author
P
N
$\langle e, \langle e, t\rangle\rangle$
e
$\lambda y \lambda x[\mathit{from}(x, y)]$
i
from
india

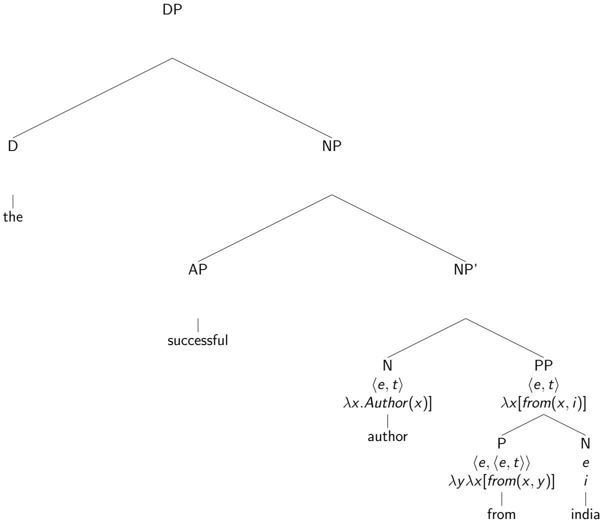
Florian vs Sandra: *the successful author from India*

a.


DP
D
the
NP
AP
successful
NP'
N
author
PP
$\langle e, t\rangle$
$\lambda x[\mathit{from}(x, i)]$
P
$\langle e, \langle e, t\rangle\rangle$
$\lambda y \lambda x[\mathit{from}(x, y)]$
from
N
e
i
india

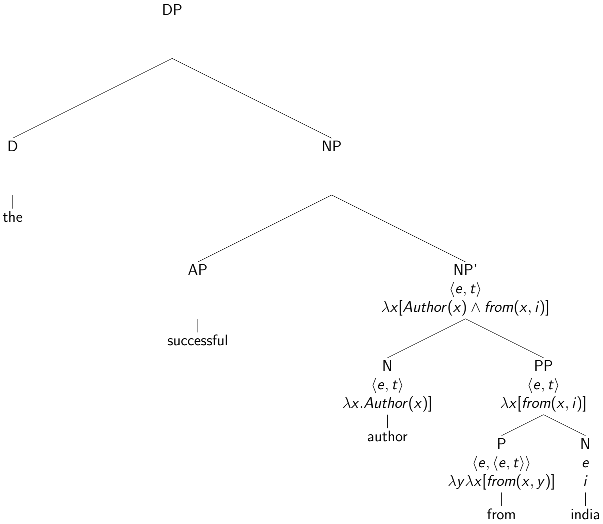
Florian vs Sandra: *the successful author from India*

a.



Florian vs Sandra: *the successful author from India*

a.



DP

D

the

NP

AP

successful

NP'

$\langle e, t\rangle$

$\lambda x[Author(x) \wedge from(x, i)]$

N

$\langle e, t\rangle$

$\lambda x.Author(x)]$

author

PP

$\langle e, t\rangle$

$\lambda x[from(x, i)]$

P

$\langle e, \langle e, t\rangle\rangle$

$\lambda y\lambda x[from(x, y)]$

from

N

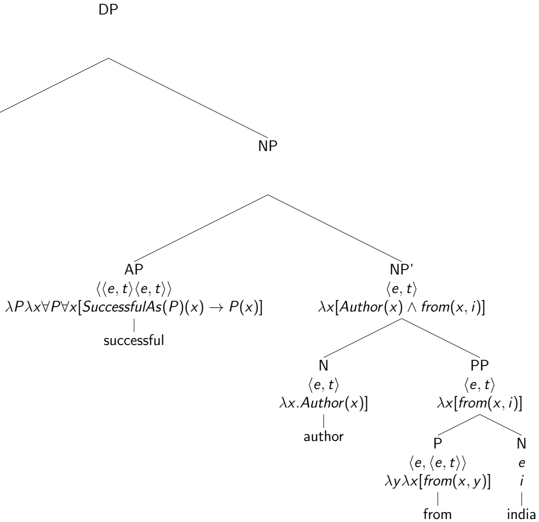
e

i

india

Florian vs Sandra: *the successful author from India*

a.



DP

D

the

NP

AP

$\langle\langle e,t\rangle\langle e,t\rangle\rangle$

$\lambda P\lambda x\forall P\forall x[SuccessfulAs(P)(x) \rightarrow P(x)]$

successful

NP'

$\langle e,t\rangle$

$\lambda x[Author(x) \wedge from(x,i)]$

N

$\langle e,t\rangle$

$\lambda x.Author(x)]$

author

PP

$\langle e,t\rangle$

$\lambda x[from(x,i)]$

P

$\langle e,\langle e,t\rangle\rangle$

$\lambda y\lambda x[from(x,y)]$

from

N

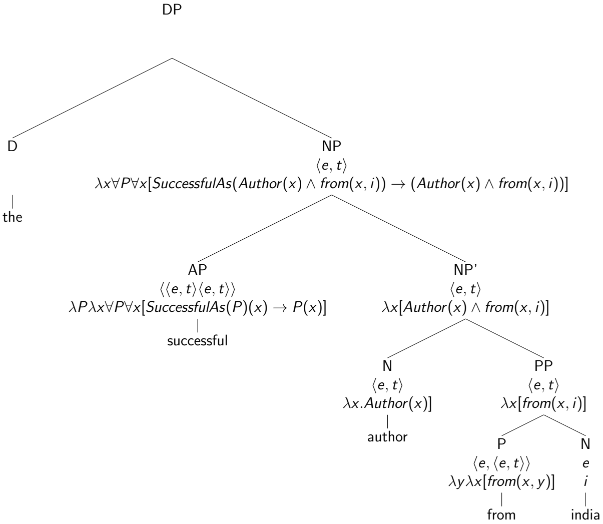
e

i

india

Florian vs Sandra: *the successful author from India*

a.



DP

D

the

NP

$\langle e, t \rangle$

$\lambda x \forall P \forall x[SuccessfulAs(Author(x) \wedge from(x, i)) \rightarrow (Author(x) \wedge from(x, i))]$

AP

$\langle \langle e, t \rangle \langle e, t \rangle \rangle$

$\lambda P \lambda x \forall P \forall x[SuccessfulAs(P)(x) \rightarrow P(x)]$

successful

NP'

$\langle e, t \rangle$

$\lambda x[Author(x) \wedge from(x, i)]$

N

$\langle e, t \rangle$

$\lambda x.Author(x)]$

author

PP

$\langle e, t \rangle$

$\lambda x[from(x, i)]$

P

$\langle e, \langle e, t \rangle \rangle$

$\lambda y \lambda x[from(x, y)]$

from

N

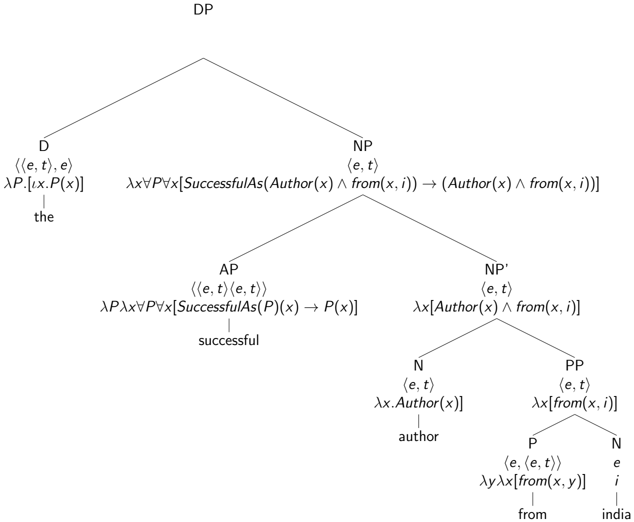
e

i

india

Florian vs Sandra: *the successful author from India*

a.



DP

D

$\langle\langle e,t\rangle,e\rangle$

$\lambda P.[\iota x.P(x)]$

the

NP

$\langle e,t\rangle$

$\lambda x\forall P\forall x[SuccessfulAs(Author(x) \wedge from(x,i)) \rightarrow (Author(x) \wedge from(x,i))]$

AP

$\langle\langle e,t\rangle\langle e,t\rangle\rangle$

$\lambda P\lambda x\forall P\forall x[SuccessfulAs(P)(x) \rightarrow P(x)]$

successful

NP'

$\langle e,t\rangle$

$\lambda x[Author(x) \wedge from(x,i)]$

N

$\langle e,t\rangle$

$\lambda x.Author(x)]$

author

PP

$\langle e,t\rangle$

$\lambda x[from(x,i)]$

P

$\langle e,\langle e,t\rangle\rangle$

$\lambda y\lambda x[from(x,y)]$

from

N

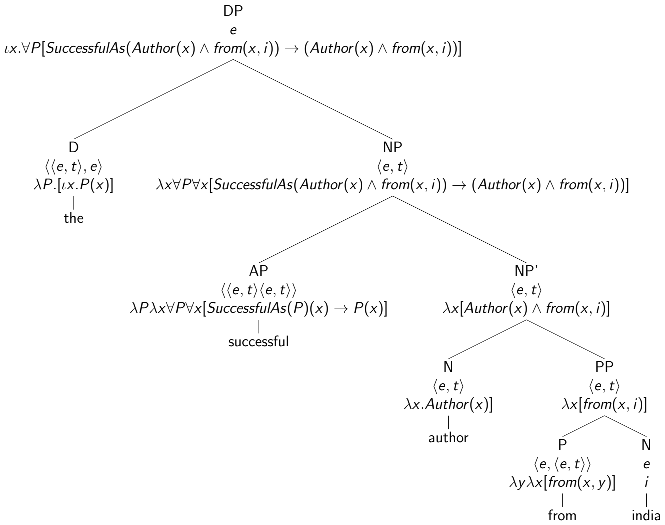
e

i

india

Florian vs Sandra: *the successful author from India*

a.



DP

e

$\iota x.\forall P[SuccessfulAs(Author(x) \wedge from(x,i)) \rightarrow (Author(x) \wedge from(x,i))]$

D

$\langle\langle e,t\rangle,e\rangle$

$\lambda P.[\iota x.P(x)]$

the

NP

$\langle e,t\rangle$

$\lambda x \forall P \forall x[SuccessfulAs(Author(x) \wedge from(x,i)) \rightarrow (Author(x) \wedge from(x,i))]$

AP

$\langle\langle e,t\rangle\langle e,t\rangle\rangle$

$\lambda P \lambda x \forall P \forall x[SuccessfulAs(P)(x) \rightarrow P(x)]$

successful

NP'

$\langle e,t\rangle$

$\lambda x[Author(x) \wedge from(x,i)]$

N

$\langle e,t\rangle$

$\lambda x.Author(x)]$

author

PP

$\langle e,t\rangle$

$\lambda x[from(x,i)]$

P

$\langle e,\langle e,t\rangle\rangle$

$\lambda y \lambda x[from(x,y)]$

from

N

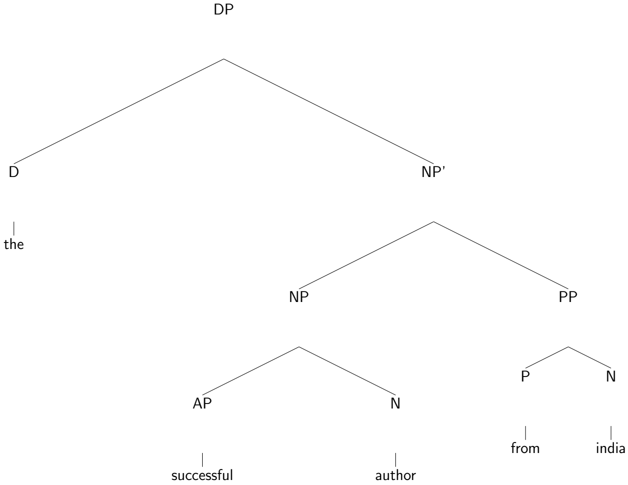
e

i

india

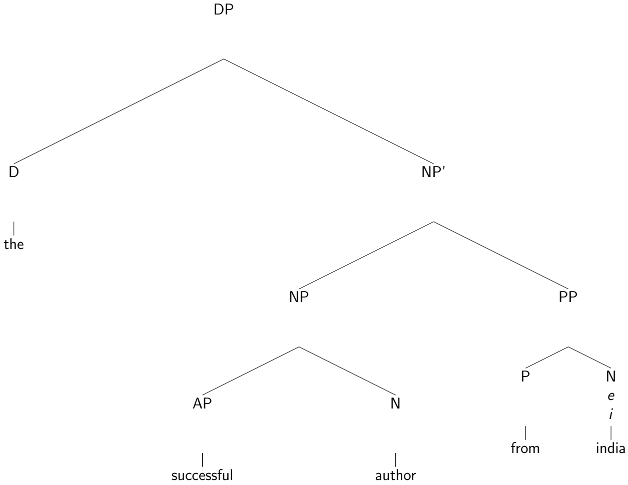
Florian vs Sandra: *the successful author from India*

b.



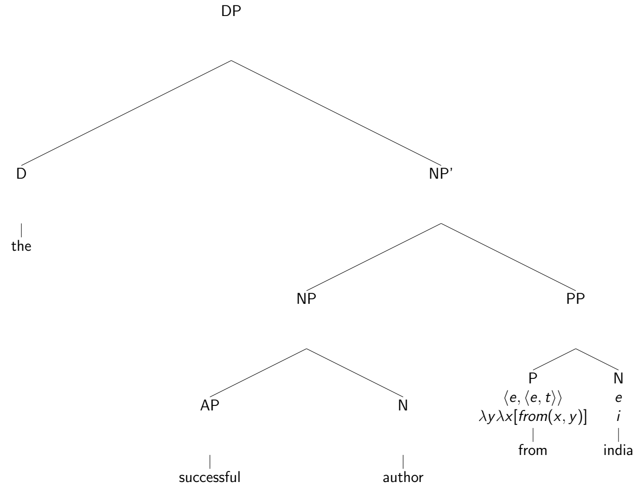
Florian vs Sandra: *the successful author from India*

b.



Florian vs Sandra: *the successful author from India*

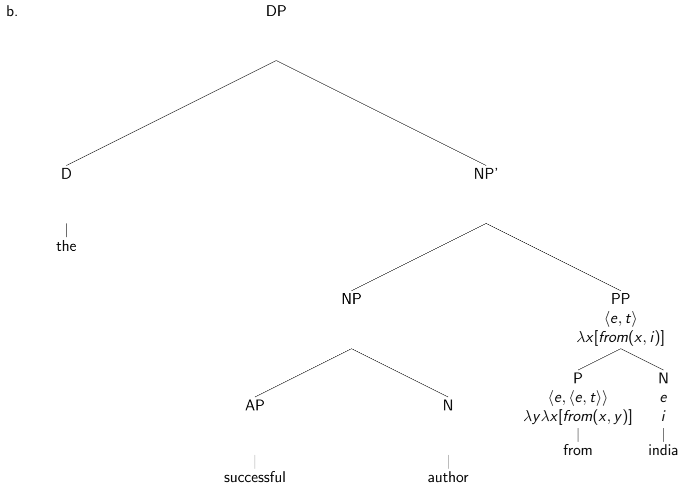
b.



DP
D
the
NP'
NP
AP
successful
N
author
PP
P
$\langle e, \langle e, t\rangle\rangle$
$\lambda y \lambda x[\mathit{from}(x, y)]$
from
N
e
i
india

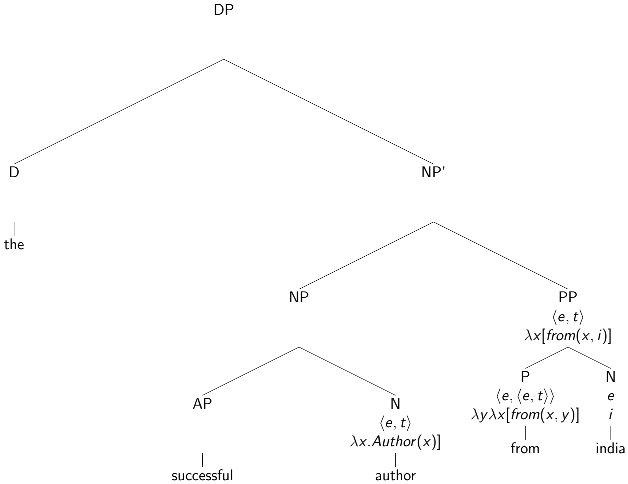
Florian vs Sandra: *the successful author from India*

b.



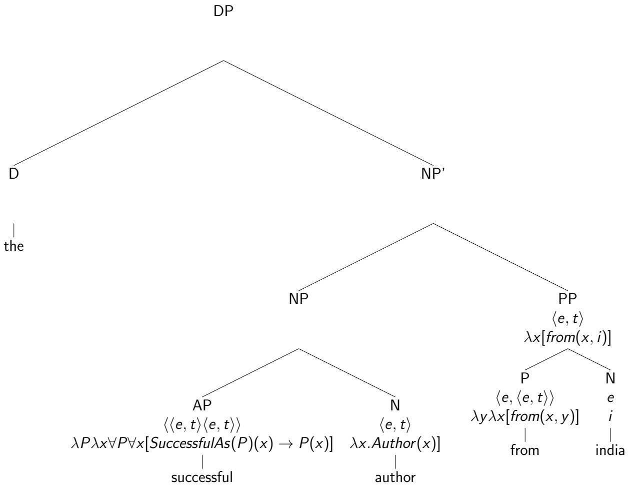
Florian vs Sandra: *the successful author from India*

b.



Florian vs Sandra: *the successful author from India*

b.



DP

D

the

NP'

NP

AP

$\langle\langle e,t\rangle\langle e,t\rangle\rangle$

$\lambda P \lambda x \forall P \forall x[\mathit{SuccessfulAs}(P)(x) \rightarrow P(x)]$

successful

N

$\langle e,t\rangle$

$\lambda x.\mathit{Author}(x)]$

author

PP

$\langle e,t\rangle$

$\lambda x[\mathit{from}(x,i)]$

P

$\langle e,\langle e,t\rangle\rangle$

$\lambda y \lambda x[\mathit{from}(x,y)]$

from

N

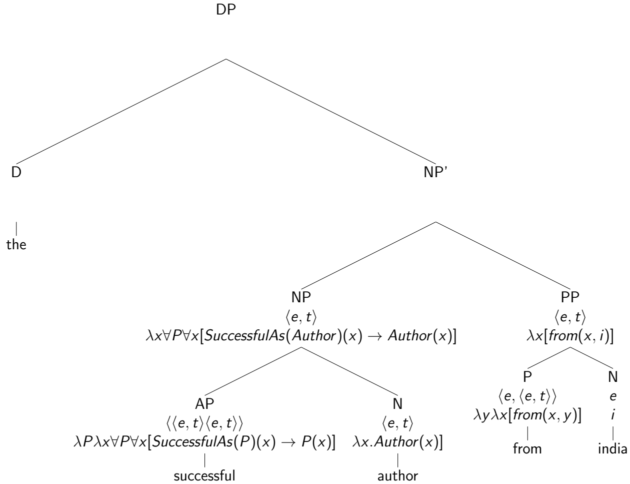
e

i

india

Florian vs Sandra: *the successful author from India*

b.



DP

D

the

NP'

NP

$\langle e, t\rangle$

$\lambda x \forall P \forall x[\mathit{SuccessfulAs}(\mathit{Author})(x) \rightarrow \mathit{Author}(x)]$

AP

$\langle\langle e, t\rangle\langle e, t\rangle\rangle$

$\lambda P \lambda x \forall P \forall x[\mathit{SuccessfulAs}(P)(x) \rightarrow P(x)]$

successful

N

$\langle e, t\rangle$

$\lambda x.\mathit{Author}(x)]$

author

PP

$\langle e, t\rangle$

$\lambda x[\mathit{from}(x, i)]$

P

$\langle e, \langle e, t\rangle\rangle$

$\lambda y \lambda x[\mathit{from}(x, y)]$

from

N

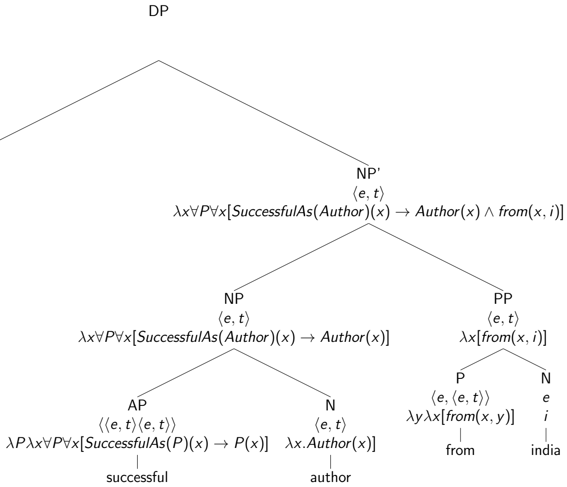
e

i

india

Florian vs Sandra: *the successful author from India*

b.



DP

D

the

NP'

$\langle e, t\rangle$

$\lambda x \forall P \forall x[SuccessfulAs(Author)(x) \rightarrow Author(x) \wedge from(x, i)]$

NP

$\langle e, t\rangle$

$\lambda x \forall P \forall x[SuccessfulAs(Author)(x) \rightarrow Author(x)]$

PP

$\langle e, t\rangle$

$\lambda x[from(x, i)]$

AP

$\langle\langle e, t\rangle\langle e, t\rangle\rangle$

$\lambda P \lambda x \forall P \forall x[SuccessfulAs(P)(x) \rightarrow P(x)]$

successful

N

$\langle e, t\rangle$

$\lambda x.Author(x)]$

author

P

$\langle e, \langle e, t\rangle\rangle$

$\lambda y \lambda x[from(x, y)]$

from

N

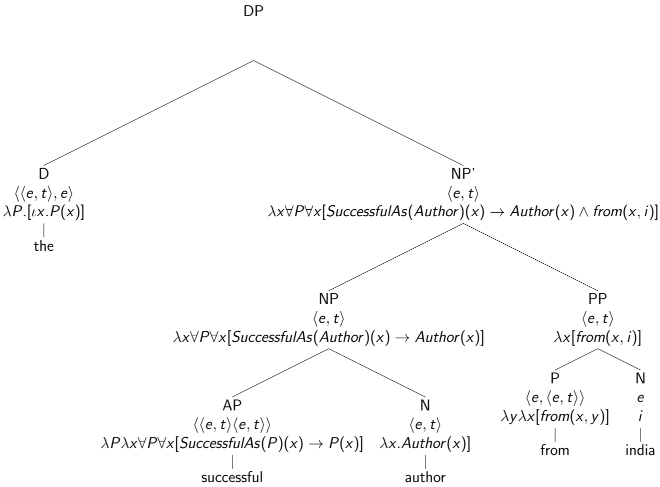
e

i

india

Florian vs Sandra: *the successful author from India*

b.



DP

D

$\langle\langle e,t\rangle,e\rangle$

$\lambda P.[\iota x.P(x)]$

the

NP'

$\langle e,t\rangle$

$\lambda x\forall P\forall x[SuccessfulAs(Author)(x) \rightarrow Author(x) \wedge from(x,i)]$

NP

$\langle e,t\rangle$

$\lambda x\forall P\forall x[SuccessfulAs(Author)(x) \rightarrow Author(x)]$

AP

$\langle\langle e,t\rangle\langle e,t\rangle\rangle$

$\lambda P\lambda x\forall P\forall x[SuccessfulAs(P)(x) \rightarrow P(x)]$

successful

N

$\langle e,t\rangle$

$\lambda x.Author(x)]$

author

PP

$\langle e,t\rangle$

$\lambda x[from(x,i)]$

P

$\langle e,\langle e,t\rangle\rangle$

$\lambda y\lambda x[from(x,y)]$

from

N

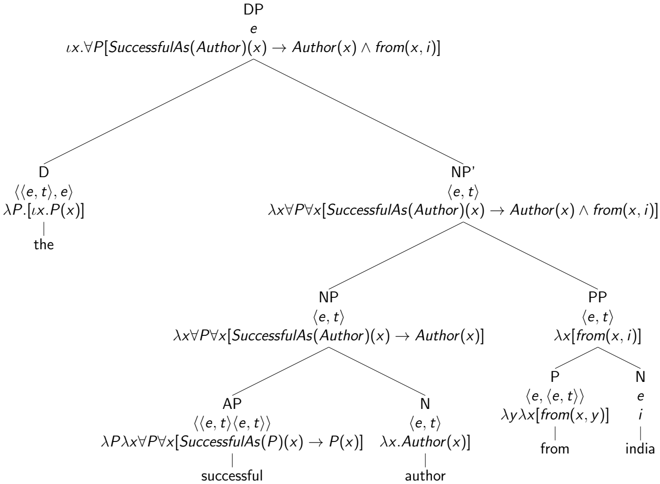
e

i

india

Florian vs Sandra: *the successful author from India*

b.



DP

e

$\iota x.\forall P[SuccessfulAs(Author)(x) \rightarrow Author(x) \wedge from(x, i)]$

D

$\langle\langle e, t\rangle, e\rangle$

$\lambda P.[\iota x.P(x)]$

the

NP'

$\langle e, t\rangle$

$\lambda x \forall P \forall x[SuccessfulAs(Author)(x) \rightarrow Author(x) \wedge from(x, i)]$

NP

$\langle e, t\rangle$

$\lambda x \forall P \forall x[SuccessfulAs(Author)(x) \rightarrow Author(x)]$

PP

$\langle e, t\rangle$

$\lambda x[from(x, i)]$

AP

$\langle\langle e, t\rangle\langle e, t\rangle\rangle$

$\lambda P \lambda x \forall P \forall x[SuccessfulAs(P)(x) \rightarrow P(x)]$

successful

N

$\langle e, t\rangle$

$\lambda x.Author(x)]$

author

P

$\langle e, \langle e, t\rangle\rangle$

$\lambda y \lambda x[from(x, y)]$

from

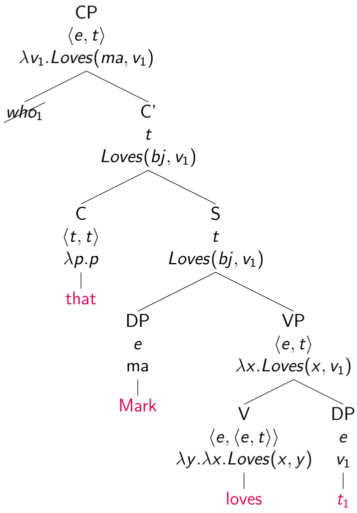
N

e

i

india

Topic 6: Beyond Function Application (Relative Clauses)



Topic 6: Beyond Function Application (Quantifier Raising)

Right Translation

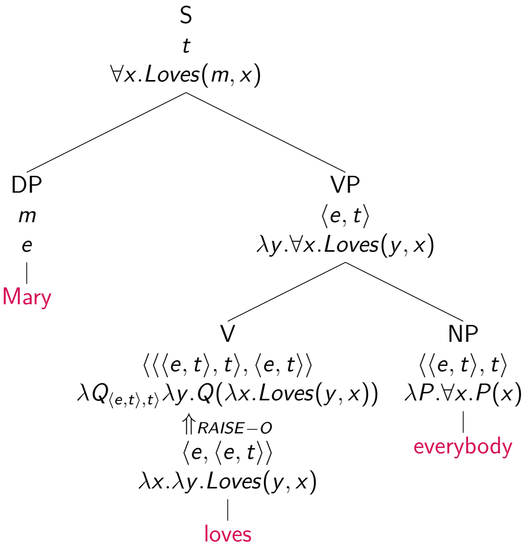
everybody $\rightsquigarrow \lambda P \forall x.P(x)$

loves $\rightsquigarrow \forall x.Loves(m, x)$

$[\lambda P \forall x.P(x)](\lambda x.Loves(m, x)) \equiv \forall x.Loves(m, x)$

Solution: QUANTIFIER RAISING: syntactic transformation that moves a quantifier, an expression of type $\langle\langle e, t\rangle, t\rangle$, to a position in the tree where it can be interpreted, and leaves a DP trace in its previous position.

Topic 6: Beyond Function Application (Type Shifting: RAISE-O)



S

t

$\forall x.Loves(m,x)$

DP

m

e

Mary

VP

$\langle e,t\rangle$

$\lambda y.\forall x.Loves(y,x)$

V

$\langle\langle\langle e,t\rangle,t\rangle,\langle e,t\rangle\rangle$

$\lambda Q_{\langle\langle e,t\rangle,t\rangle}\lambda y.Q(\lambda x.Loves(y,x))$

$\Uparrow_{RAISE-O}$

$\langle e,\langle e,t\rangle\rangle$

$\lambda x.\lambda y.Loves(y,x)$

loves

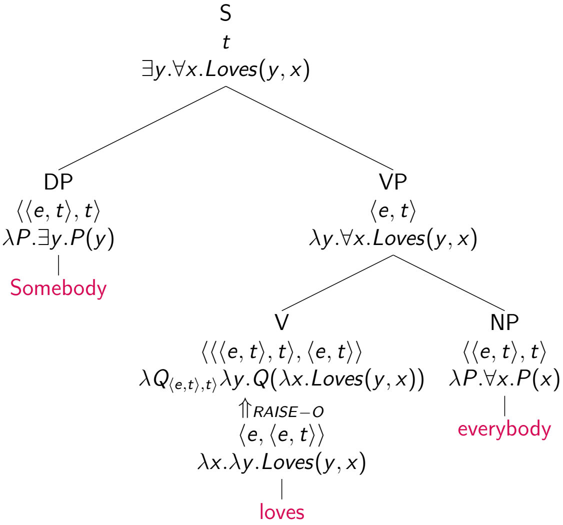
NP

$\langle\langle e,t\rangle,t\rangle$

$\lambda P.\forall x.P(x)$

everybody

Topic 6: Beyond Function Application (Two Quantifiers)



S

t

$\exists y.\forall x.Loves(y,x)$

DP

$\langle\langle e,t\rangle,t\rangle$

$\lambda P.\exists y.P(y)$

Somebody

VP

$\langle e,t\rangle$

$\lambda y.\forall x.Loves(y,x)$

V

$\langle\langle\langle e,t\rangle,t\rangle,\langle e,t\rangle\rangle$

$\lambda Q_{\langle e,t\rangle,t\rangle}\lambda y.Q(\lambda x.Loves(y,x))$

$\Uparrow_{RAISE-O}$

$\langle e,\langle e,t\rangle\rangle$

$\lambda x.\lambda y.Loves(y,x)$

loves

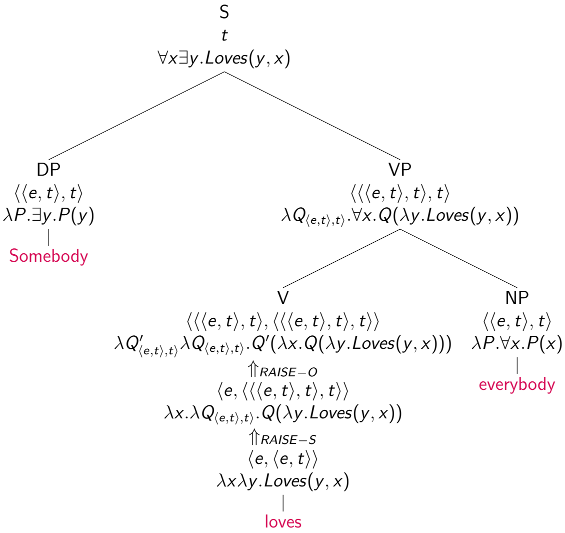
NP

$\langle\langle e,t\rangle,t\rangle$

$\lambda P.\forall x.P(x)$

everybody

Topic 6: Beyond Function Application (Type Shifting: RAISE-S)



S

t

$\forall x \exists y.Loves(y, x)$

DP

$\langle\langle e, t\rangle, t\rangle$

$\lambda P.\exists y.P(y)$

Somebody

VP

$\langle\langle\langle e, t\rangle, t\rangle, t\rangle$

$\lambda Q_{\langle e,t\rangle,t\rangle}.\forall x.Q(\lambda y.Loves(y, x))$

V

$\langle\langle\langle e, t\rangle, t\rangle, \langle\langle\langle e, t\rangle, t\rangle, t\rangle\rangle$

$\lambda Q'_{\langle e,t\rangle,t\rangle}\lambda Q_{\langle e,t\rangle,t\rangle}.Q'(\lambda x.Q(\lambda y.Loves(y, x)))$

$\Uparrow_{RAISE-O}$

$\langle e, \langle\langle\langle e, t\rangle, t\rangle, t\rangle\rangle$

$\lambda x.\lambda Q_{\langle e,t\rangle,t\rangle}.Q(\lambda y.Loves(y, x))$

$\Uparrow_{RAISE-S}$

$\langle e, \langle e, t\rangle\rangle$

$\lambda x \lambda y.Loves(y, x)$

loves

NP

$\langle\langle e, t\rangle, t\rangle$

$\lambda P.\forall x.P(x)$

everybody

Topic 7: Presupposition (triggers)

There are certain words/constructions that signal presupposition.

Example

Every man kissed the woman who loved him. >> There is at least one man and he kissed the woman who loved him.

Neither theory can explain why sentences whose presuppositions are not satisfied are odd. >> There are exactly two theories. (*quantifiers*)

John's daughter will come. >> John has a daughter. (*possessives*)

John knows that Mary hates Bill. >> Mary hates Bill (*cognitive factive verbs*).

John is happy that Mary agrees to marry him. >> Mary agrees to marry John. (*emotive factive verbs*)

Mary bakes cookies again. >> Mary has baked cookies before. (*additive adverbs*)

It was John who broke the computer. >> Someone broke the computer. (*Clefts*)

The student is smart. >> There is an unique student in the context. (*definite determiners*)

Topic 7: Presupposition (Defineteness Conditions)

Definite determiner ‘the’ is one of the presupposition trigers, what about others? Both, neither, every, etc.

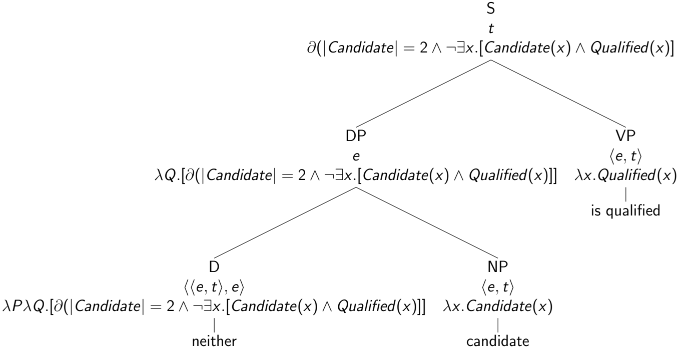
∂ operator is defined like $\partial(|P| = 2)$ reads ‘presupposing that there are exactly two Ps’.

neither $\rightsquigarrow \lambda P \lambda Q.[\partial(|P| = 2) \wedge \neg \exists x.[P(x) \wedge Q(x)]]$
every $\rightsquigarrow \lambda P \lambda Q.[\partial(\exists x.P(x)) \wedge \forall x.[P(x) \rightarrow Q(x)]]$

Topic 7: Presupposition (Defineteness Conditions)

Example

Neither candidate is qualified.


S
t
$\partial(|Candidate| = 2 \wedge \neg\exists x.[Candidate(x) \wedge Qualified(x)]$
DP
e
$\lambda Q.[\partial(|Candidate| = 2 \wedge \neg\exists x.[Candidate(x) \wedge Qualified(x)]]$
VP
$\langle e, t\rangle$
$\lambda x.Qualified(x)$
is qualified
D
$\langle\langle e, t\rangle, e\rangle$
$\lambda P \lambda Q.[\partial(|Candidate| = 2 \wedge \neg\exists x.[Candidate(x) \wedge Qualified(x)]]$
neither
NP
$\langle e, t\rangle$
$\lambda x.Candidate(x)$
candidate

Topic 7: Presupposition (Weak vs Strong Kleene Connectives)

Weak Star Idea: We see as a $\#$ “contaminating” (or nonsense) value, which does not allow us to deduce anything if there is a presupposition failure somewhere.
Strong Star Idea: Idea: We see $\#$ in one argument as “ignorance” (unknown) - it still allows us to deduce the result from the value of the other argument.

Topic 7: Presupposition (Identity)

The King of France is the Grand Sultan of Germany.

LaPierre (1992) defines identity between two terms as follows:

- if neither α nor β denotes the undefined individual, then $[\![\alpha = \beta]\!]^{M,g} = 1$ if $[\![\alpha]\!]^{M,g} = [\![\beta]\!]^{M,g}$, and 0 otherwise.
- If one α or β denotes the undefined individual, then $[\![\alpha = \beta]\!]^{M,g} = 0$
- If both denote the undefined individual, then $[\![\alpha = \beta]\!]^{M,g} = \#$ is undefined (not enough is "known" about the objects to determine that they are the same or distinct).

Topic 7: Presupposition (Predication with Undefined Individuals)

Semantic Rule: Existence Predicate

$[\![Exists(\alpha)]\!]^{M,g} = 1$ if $[\![\alpha]\!]^{M,g} \neq \#_e$ and 0 otherwise

Predict truth value of ‘The Golden Mountain does not exist’

Topic 8:

Topic 9: Aspect & Tense (durative vs terminative)

A situation, process, action etc. or the verb phrase, sentence, etc, expressing this situation, etc, has the property iff

- It is directed toward attaining a goal or limit at which the action exhausts itself and passed into something else (Andersson, 1972)
- It leads up to a well-defined point behind which the process cannot continue (Comrie, 1976)
- has actual or potential terminal point (Dahl, 1981)

A terminal point t is defined such that

1. if t is reached, the process cannot continue
2. t will be reached in the normal course of events ($=$ if nothing unexpected intervenes)
3. t will be reached in all possible courses of events.

Topic 9: Aspect & Tense (Aktionsarten)

1. **States** are static, extended in time, and lack a natural end point.
2. **Activities** are like states except they typically involve or lead to some kind of change.
3. **Accomplishments** are like activities except they have a natural end point.
4. **Achievements** are like accomplishments except they are punctual rather than extended in time.

State	$V(x_1, ..., x_n)$
Activity	$DO(x_1, V(x_1, ..., x_n))$
Accomplishment	$DO(x_1, V(x_1, ..., x_n)) CAUSE (BECOME\ V(x_1, ..., x_n))$
Achievement	$(BECOME\ V(x_1, ..., x_n))$

Topic 9: Aspect &Tense (Coercion)

Example

a. Mary walked miles $V(Act)$

b. Mary walked three miles $V(Acc)(=\uparrow V(Act))$

c. My program ran for a few minutes. $V(Act)$

d. My program ran in less than four minutes (this morning). $V(Acc)(=\uparrow V(Act))$

e. Suddenly, I knew the answer. $V(Ach)(=\uparrow V(State))$

f. John played the sonata for about eight hours. $V(Acc)(=\uparrow V(Act))$

g. For months, the train arrived late. $V(State)(=\uparrow V(Ach))$

Topic 9: Aspect &Tense (inner aspectuality)

Example

(a) Mary walked three miles $V[+Addto] + NP(int)[+SQA] \rightsquigarrow [+T(VP)]$

(b) Mary walked miles $V[+Addto] + NP(int)[-SQA] \rightsquigarrow [-T(VP)]$

Calculate the plus/minus value of terminativity at the sentence level:

Example

a.[S Mary [VP walk [NP three miles]]]
$[+Ts[+SQA]$ $[+Tvp[+ADDTO]$ $[+SQA]]]$ (terminative)

b. [S Mary [VP walk [NP miles]]]
$[-Ts[+SQA]$ $[-Tvp[+ADDTO]$ $[-SQA]]]$ (durative)

c. [S Children [VP walk [NP three miles]]]
$[-Ts[-SQA]$ $[+Tvp[+ADDTO]$ $[+SQA]]]$ (durative)

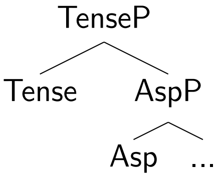
d.[S Mary [VP save [NP three miles]]]
$[-Ts[+SQA]$ $[-Tvp[-ADDTO]$ $[+SQA]]]$ (durative)

Topic 9: Aspect &Tense (Viewpoint Aspect)

	Temporal relations	Tense category	Traditional label	Example
1.	E–R–S	Anterior Past	Past Perfect	I had passed the exam by the end of the winter.
2.	E,R–S	Simple Past	Simple Past	I passed the exam.
3.a	R–E–S	Posterior Past	–	I did not know that he would win. (yesterday)
3.b	R–S,E	Posterior Past	–	I did not know that he would be here. (right now)
3.c	R–S–E	Posterior Past	–	I did not know that he would come. (tomorrow)
4.	E–S,R	Anterior Present	Present Perfect	I have passed the exam.
5.	S,R,E	Simple Present	Present	I see John.
6.	S,R–E	Posterior Present	Simple Future	I shall see John.
7a	S–E–R	Anterior Future	Future Perfect	I will have passed the exam by the end of the winter.
7.b	S,E–R	Anterior Future	Future Perfect	John will have fixed the car by tonight.(already repaired)
7.c	E–S–R	Anterior Future	Future Perfect	I will have fixed the car by tonight. (just repairing it)
8.	S–R,E	Simple Future	Simple future	I will see John tomorrow.
9.	S–R–E	Posterior Future		I shall be going to see him.

Topic 9: Aspect &Tense (Theory of Tense: formalization)

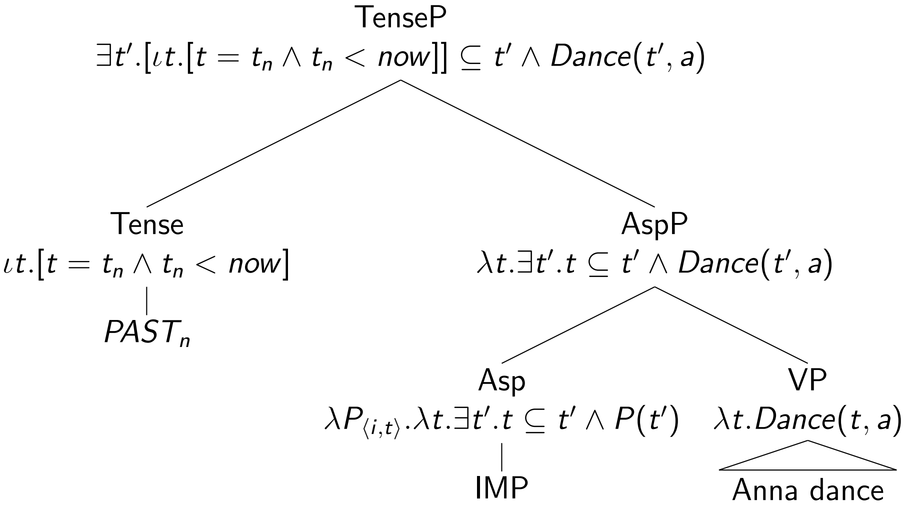
[*Tense*[*Aspect* $*$ [*eventuality description*] where the Kleene star $*$ indicates zero, one or more operations.



$PERF \leadsto \lambda P_{\langle i,t\rangle}.\lambda t.\exists t'.t' \subseteq t \wedge P(t')$
$IMP \leadsto \lambda P_{\langle i,t\rangle}.\lambda t.\exists t'.t \subseteq t' \wedge P(t')$

PAST $PAST_n \leadsto \iota t.[t = t_n \wedge t_n < now]$
PRESENT $PRESENT \leadsto now$
WOLL $\lambda P_{\langle i,t\rangle}.\lambda t.\exists t'.t < t' \wedge P(t')$

Topic 9: Aspect & Tense (Derivation for past (imperfective))



TenseP

$\exists t'.[\iota t.[t = t_n \wedge t_n < now]] \subseteq t' \wedge Dance(t', a)$

Tense

$\iota t.[t = t_n \wedge t_n < now]$

$PAST_n$

AspP

$\lambda t.\exists t'.t \subseteq t' \wedge Dance(t', a)$

Asp

$\lambda P_{\langle i,t \rangle}.\lambda t.\exists t'.t \subseteq t' \wedge P(t')$

IMP

VP

$\lambda t.Dance(t, a)$

Anna dance

Topic 10: Event Semantics (diamond entailments)

Example 2

(1) Brutus was a famous Roman politician.
(2) Brutus was a Roman politician.
(3) Brutus was a famous politician.
(4) Brutus was a politician.

Example 2

(1) Brutus stabbed Caesar at noon on the forum.
(2) Brutus stabbed Caesar on the forum.
(3) Brutus stabbed Caesar at noon.
(4) Brutus stabbed Caesar.

Event Semantics: Advantages of the Neo-Davidsonian approach

- Makes it easier to state generalizations across the categories of nouns and verbs, and to place constraints on thematic roles
- Good for formulating analyses without committing to an argument/adjunct distinction
- Lends itself to a natural compositional process in terms of intersection with an existential quantifier at the end

Example

a. $[\![[agent]]\!] = \lambda x \lambda e[agent(e) = x]$

b. $[\![[theme]]\!] = \lambda y \lambda e[agent(e) = y]$

c. $[\![stab]\!] = \lambda e[stab(e)]$

d. $[\![[agent]Brutus]\!] = \lambda e[agent(e) = brutus]$

e. $[\![[theme]Caesar]\!] = \lambda e[agent(e) = caesar]$

f. $[\![$ Brutus stab Caesar $]\!] = (c) \cap (d) \cap (e)$ *(sentence radical)*

g. $[\![$ Brutus stab Caesar $]\!] = \exists e\ e \in (c) \cap (d) \cap (e)$ *(full sentence)*

Topic 10: Event Semantics (composition)

Predicates

stab $\rightsquigarrow \lambda e.Stab(e)$

butter $\rightsquigarrow \lambda e.Butter(e)$

Syntax

DP $\rightarrow \theta$ DP

Lexicon

θ: [agent], [theme], [instrument], [recipient], [goal], [location], [time], ...

theta Mapping

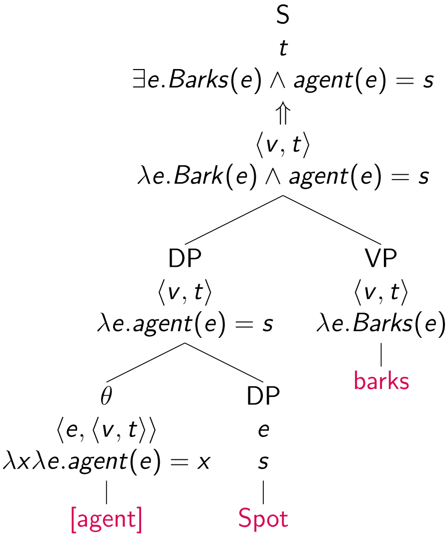
[agent] $\rightsquigarrow \lambda x \lambda e.agent(e) = x$

[theme] $\rightsquigarrow \lambda x \lambda e.theme(e) = x$

[instrument] $\rightsquigarrow \lambda x \lambda e.instrument(e) = x$

...

Topic 10: Event Semantics (existential closure)



S

t

$\exists e.Barks(e) \wedge agent(e) = s$

$\Uparrow$

$\langle v, t\rangle$

$\lambda e.Bark(e) \wedge agent(e) = s$

DP

$\langle v, t\rangle$

$\lambda e.agent(e) = s$

VP

$\langle v, t\rangle$

$\lambda e.Barks(e)$

barks

θ

$\langle e, \langle v, t\rangle\rangle$

$\lambda x \lambda e.agent(e) = x$

[agent]

DP

e

s

Spot

Topic 10: Event Semantics (verbs as event quantifiers)

a. $\lambda f \exists e.[Barks(e) \wedge agent(e) = s \wedge f(e)](\lambda e.true)$
b. $\exists e.[Barks(e) \wedge agent(e) = s \wedge (\lambda e.true)(e)]$
c. $\exists e.[Barks(e) \wedge agent(e) = s \wedge true]$
d. $\exists e.[Barks(e) \wedge agent(e) = s]$

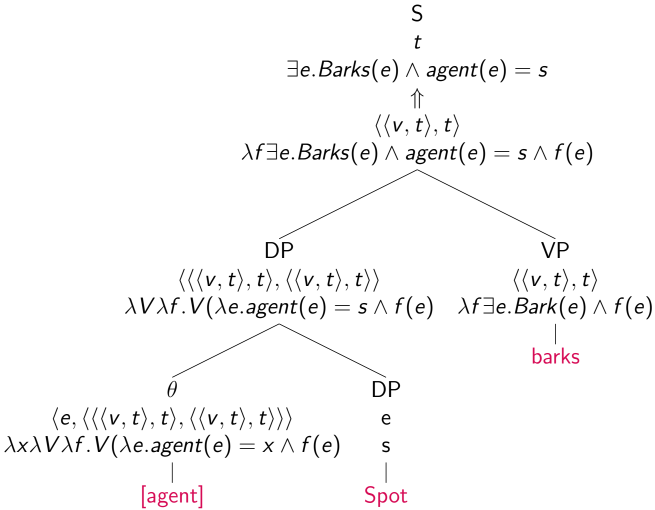
Type-Shifting Rule 6: Quantifier Closure

if $\alpha \rightsquigarrow \alpha'$, where α' is of a category $\langle\langle v, t\rangle, t\rangle$, then:

$$\alpha \rightsquigarrow \alpha'(\lambda e.true)$$

as well.

Topic 10: Event Semantics (verbs as event quantifiers)


S
t
$\exists e.Barks(e) \wedge agent(e) = s$
$\Uparrow$
$\langle\langle v, t\rangle, t\rangle$
$\lambda f \exists e.Barks(e) \wedge agent(e) = s \wedge f(e)$
DP
$\langle\langle\langle v, t\rangle, t\rangle, \langle\langle v, t\rangle, t\rangle\rangle$
$\lambda V \lambda f.V(\lambda e.agent(e) = s \wedge f(e)$
VP
$\langle\langle v, t\rangle, t\rangle$
$\lambda f \exists e.Bark(e) \wedge f(e)$
barks
θ
$\langle e, \langle\langle\langle v, t\rangle, t\rangle, \langle\langle v, t\rangle, t\rangle\rangle\rangle$
$\lambda x \lambda V \lambda f.V(\lambda e.agent(e) = x \wedge f(e)$
[agent]
DP
e
s
Spot

Topic 10: Event Semantics (other quantifiers)

- Quantificational Noun Phrase
- Quantificational Adjuncts
- Negation and its scope