

## Scope ambiguity: preliminaries 1

- the basic problem: there is no 1:1-relation between syntax and semantics
- there are in principle three ways of coping with this mismatch
  - assume a nondeterministic 1:1-relation between syntax and semantics
  - assume a different syntactic structure for each reading
  - assume one semantic representation only for one syntactic structure
- Cooper Storage approaches (including the 1994 HPSG book) choose the first option
  - semantic representations consist of a list of quantifiers and the semantic representation proper
  - at specific points in the derivation, quantifiers may be **discharged**, i.e., applied to the semantic representation proper
  - the syntactic origin of a quantifier determines its minimal scope



## Scope ambiguity: preliminaries 2

- simplified example: reading ' $\exists\forall$ ' of (39)

(39) *Every woman loves a man*

- *a man*:  $\langle x_1, \{[\lambda Q\exists x_1(\mathbf{man}'(x_1) \wedge Q(x_1))]_1\} \rangle$
- *loves a man*:  $\langle \lambda x.\mathbf{love}'(x, x_1), \{[\lambda Q\exists x_1(\mathbf{man}'(x_1) \wedge Q(x_1))]_1\} \rangle$
- *every woman loves a man*:  
 $\langle \mathbf{love}'(x_2, x_1), \{[\lambda Q\exists x_1(\mathbf{man}'(x_1) \wedge Q(x_1))]_1, [\lambda P\forall x_2(\mathbf{woman}'(x_2) \rightarrow P(x_2))]_2\} \rangle$   
 $\langle \forall x_2(\mathbf{woman}'(x_2) \rightarrow \mathbf{love}'(x_2, x_1)), \{[\lambda Q\exists x_1(\mathbf{man}'(x_1) \wedge Q(x_1))]_1\} \rangle$   
 $\langle \exists x_1(\mathbf{man}'(x_1) \wedge \forall x_2(\mathbf{woman}'(x_2) \rightarrow \mathbf{love}'(x_2, x_1))), \{\} \rangle$



## Scope ambiguity: preliminaries 3

- Generative Grammar adopts the second choice
  - motivation for Generative Grammar: structural semantic ambiguities are taken as a diagnostic for non-surface syntactic levels
  - syntactic levels are connected by structural operations, in particular, various kinds of **movement**
  - the level of **Logical Form** (LF) is relevant for interpretation
  - readings of structurally ambiguous expressions correspond to different LF structures, e.g., for (39) (simplified):

(40)  $[_{IP} [\text{every man}]_i [\text{a woman}]_j [\text{IP } t_i \text{ loves } t_j]]]$

(41)  $[_{IP} [\text{a woman}]_j [\text{IP } [\text{every man}]_i [\text{IP } t_i \text{ loves } t_j]]]$



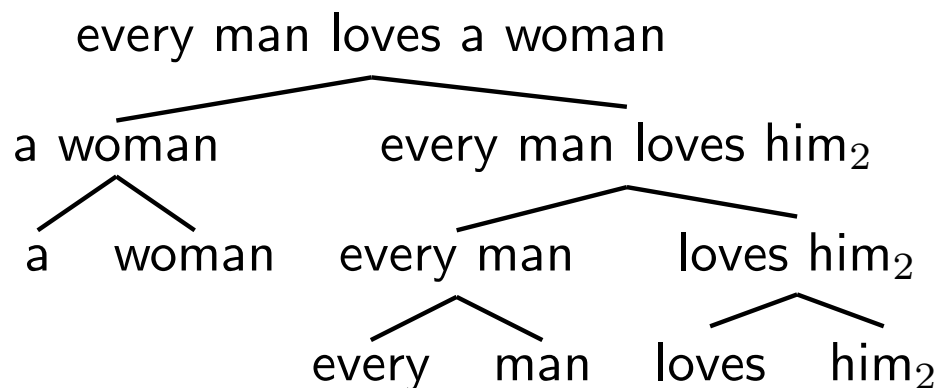
## Scope ambiguity: preliminaries 4

- traditional Montague Grammar opts for the second choice, too
  - Montague's motivation to resolve the mismatch
    - \* model interpretation as a **function** from syntax into semantics
    - \* regard semantic representation a convenient but in principle expendable step
  - the 1:1 syntax-semantics correspondence holds between syntactic derivation trees and semantic representations
  - to get additional syntactic structures, Montague introduced **quantifying in**
    - \* a destructive operation on syntactic structures
    - \* roughly, replacement of [the first occurrence of] a dummy pronoun by a quantifying NP



## Scope ambiguity: preliminaries 5

- example: reading '∃∀' of (39)



- semantic interpretation:

$$\begin{aligned}
 & \lambda P \exists x (\mathbf{woman}'(x) \wedge P(x)) \\
 & \quad (\lambda x_2 \lambda Q \forall y (\mathbf{man}'(y) \rightarrow Q(y)) \\
 & \quad \quad (\lambda x \mathbf{love}'(x, x_2))) = \\
 & \exists x (\mathbf{woman}'(x) \wedge \forall y (\mathbf{man}'(y) \rightarrow \mathbf{love}'(y, x)))
 \end{aligned}$$



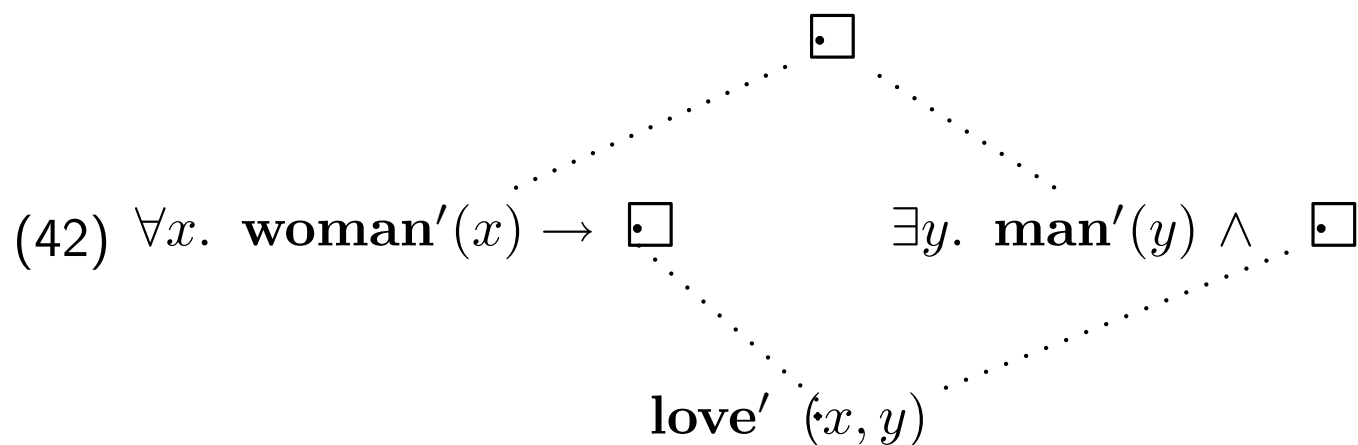
## Scope ambiguities: preliminaries 6

- semantic underspecification is the implementation of the third choice
- motivation: efficient and non-disjunctive representation of structural ambiguities
- basic idea 1: represent ambiguous expressions as the **set** of their readings
- this avoids disjunctions, but for complex expressions, enumerating this set would not be efficient (Catalan numbers)
- two ways of representing the set:
  - enumerate its elements
  - describe its elements in terms of a property  $P$  that pertains to all elements of the set and to nothing else



## Scope ambiguities: the first approach 1

- the first approach to scope ambiguity implements this second way
- the ingredients of a description of a set of  $\lambda$ -terms
  - fragments of  $\lambda$ -terms
  - holes (' $\square$ ') model not yet known parts of fragments
  - dominance relations (part-of relations) relate holes and fragments
- example: the representation of (39), a 'dominance diamond'



## Scope ambiguities: the first approach 2

- such expressions of an underspecification formalism are ‘constraints’ over object-level semantic representations, here,  $\lambda$ -terms
- object-level semantic representations that are described by a constraint are called its **solutions**
- solutions are derived from constraints by **monotonic addition** of information
  - in particular, by strengthening (im-)proper part-of relations to identity
- (42) has two solutions:
  - (43)  $\exists y.\mathbf{man}'(y) \wedge \forall x.\mathbf{woman}'(x) \rightarrow \mathbf{love}'(x, y)$
  - (44)  $\forall x.\mathbf{woman}'(x) \rightarrow \exists y.\mathbf{man}'(y) \wedge \mathbf{love}'(x, y)$
- examples: UDRT (Reyle 1993), MRS (Copestake et al. 2001), CLLS (Egg et al. 2001)



## Scope ambiguities: the second approach 1

- specify an underlying semantic representation, which is close to the syntax
- then design a (typically nondeterministic) algorithm that derives (possibly several) ‘surface’ semantic representations from the underlying representation

(45) *Every researcher of a company smiled*

- underlying representation of (45); quantifiers are represented as ‘terms’:

(46)  $\text{smile}'(\langle \forall r.\text{researcher}'(r) \wedge \text{of}'(r, \langle \exists c.\text{company}'(c) \rangle) \rangle)$

- simplified form of the algorithm
  - take the arguments of the top-level predicate and apply their terms to the predicate they are an argument of (or don't)
  - repeat if terms are nested
  - then scope any remaining terms



## Scope ambiguities: the second approach 2

- example for the application of a term to its predicate:
  - input:  $\mathbf{smile}'(\langle \forall r.\mathbf{researcher}'(r) \rangle)$
  - output:  $\forall r.\mathbf{researcher}'(r) \rightarrow \mathbf{smile}'(r)$
- for (45), there are two possibilities:
  - the first step returns the argument of the *smile*-predicate without changes

$$(47) \langle \forall r.\mathbf{researcher}'(r) \wedge \mathbf{of}'(r, \langle \exists c.\mathbf{company}'(c) \rangle) \rangle$$

- or with the term discharged

$$(48) \langle \forall r.\mathbf{researcher}'(r) \wedge \exists c.\mathbf{company}'(c) \wedge \mathbf{of}'(r, c) \rangle$$



## Scope ambiguities: the second approach 3

- applying the argument of the *smile*-predicate itself (step 3) thus returns (49) or (50):

$$(49) \quad \forall r.\mathbf{researcher}'(r) \wedge \mathbf{of}'(r, \langle \exists c.\mathbf{company}'(c) \rangle) \rightarrow \mathbf{smile}'(r)$$

$$(50) \quad \forall r.\mathbf{researcher}'(r) \wedge \exists c(\mathbf{company}'(c) \wedge \mathbf{of}'(r, c)) \rightarrow \mathbf{smile}'(r)$$

- in (49), further application of the remaining term returns:

$$(51) \quad \exists c.\mathbf{company}'(c) \wedge \forall r.\mathbf{researcher}'(r) \wedge \mathbf{of}'(r, c) \rightarrow \mathbf{smile}'(r)$$

- example: the algorithm of Hobbs and Shieber (1987)



## Scope ambiguity: challenges

- nested quantifiers (Keller 1988; Hobbs and Shieber 1987)

(52) Every researcher of a company saw most samples

- 'lowering' of quantifiers

(53) A unicorn seems to be in the garden

- quantifiers in ellipses (Hirschbühler 1982)

(54) Every linguist attended a workshop. Every computer scientist did, too

- anaphoric dependencies between quantifiers

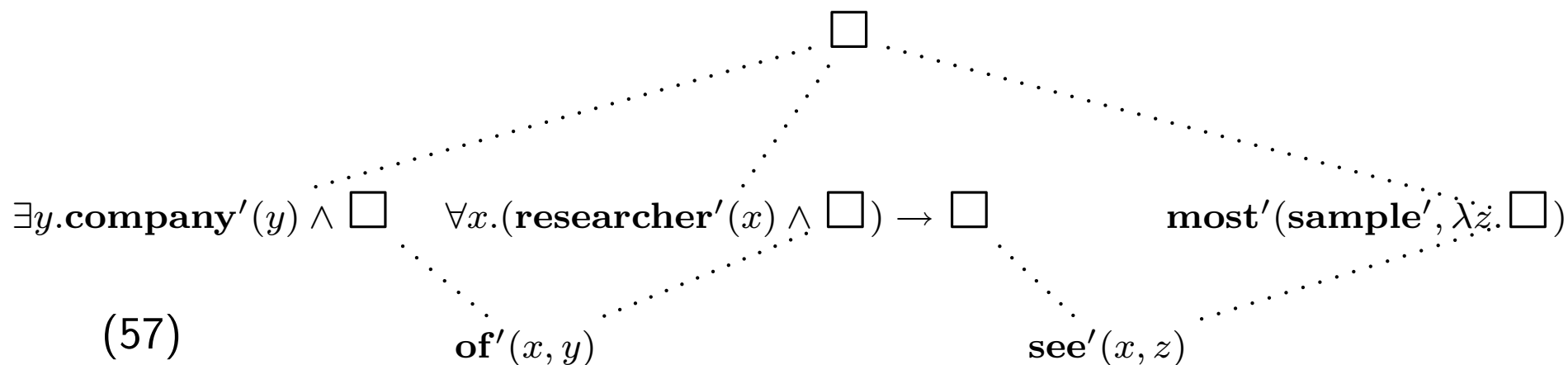
(55) [Every woman]<sub>*i*</sub> loves a man whom she<sub>*i*</sub> knows



## Scope ambiguity: nested quantifiers 1

- nested quantifiers can be handled by **dominance diamonds**, for (56) [= (52)]:

(56) Every researcher of a company saw most samples



- two readings are derivable by first scoping the *most*-fragment below the *forall*-fragment (which results in a simple dominance diamond)
- scoping the *most*- above the *forall*-fragment yields three readings (the *exists*-fragment may lie above, between, or below the other two)

## Scope ambiguity: nested quantifiers 2

- in this representation, the *most*-fragment can never be below the *forall*- but above the *exists*-fragment
- this is the unwanted sixth ( $3! = 6$ ) reading
- this solution was first worked out in Bos (1996)
- it is a declarative version of Hobbs and Shieber:
  - the quantifier must have scope over its nucleus
  - the nucleus of a quantifier is the verb or preposition that has the NP as its argument whose semantics contributes the quantifier
  - the dominance relations ensure this as well as the scoping algorithm



## Scope ambiguity: quantifier lowering 1

- in sentences like (58) [= (53)], the scope of the quantifier and the raising verb is open (*de dicto* vs. *de re* reading)

(58) A unicorn seems to be in the garden

- the quantifier must be able to get scope **below the verb**, although it is introduced in the syntax only after the VP *seems to be in the garden* has been constructed
- in Cooper Storage approaches, syntactic introduction determines minimal scope
- in approaches like CLLS or MRS, no such restriction applies
- it is the task of semantic construction to get this right



## Scope ambiguity: quantifier lowering 2

- the representation of (58):  $\Box$

$$(59) \text{ seem}'(\wedge \Box) \quad \exists x. \text{unicorn}'(x) \wedge \Box$$

$$\text{be-in}'(x, \mathbf{G})$$

- $\text{seem}'(p)$  is true in a world  $w$  iff  $p$  is true in all possible worlds where things are as they seem in  $w$
- two solutions:

$$(60) \text{ seem}'(\wedge \exists x. \text{unicorn}'(x) \wedge \text{be-in}'(x, \mathbf{G}))$$

$$(61) \exists x. \text{unicorn}'(x) \wedge \text{seem}'(\wedge (\text{be-in}'(x, \mathbf{G})))$$



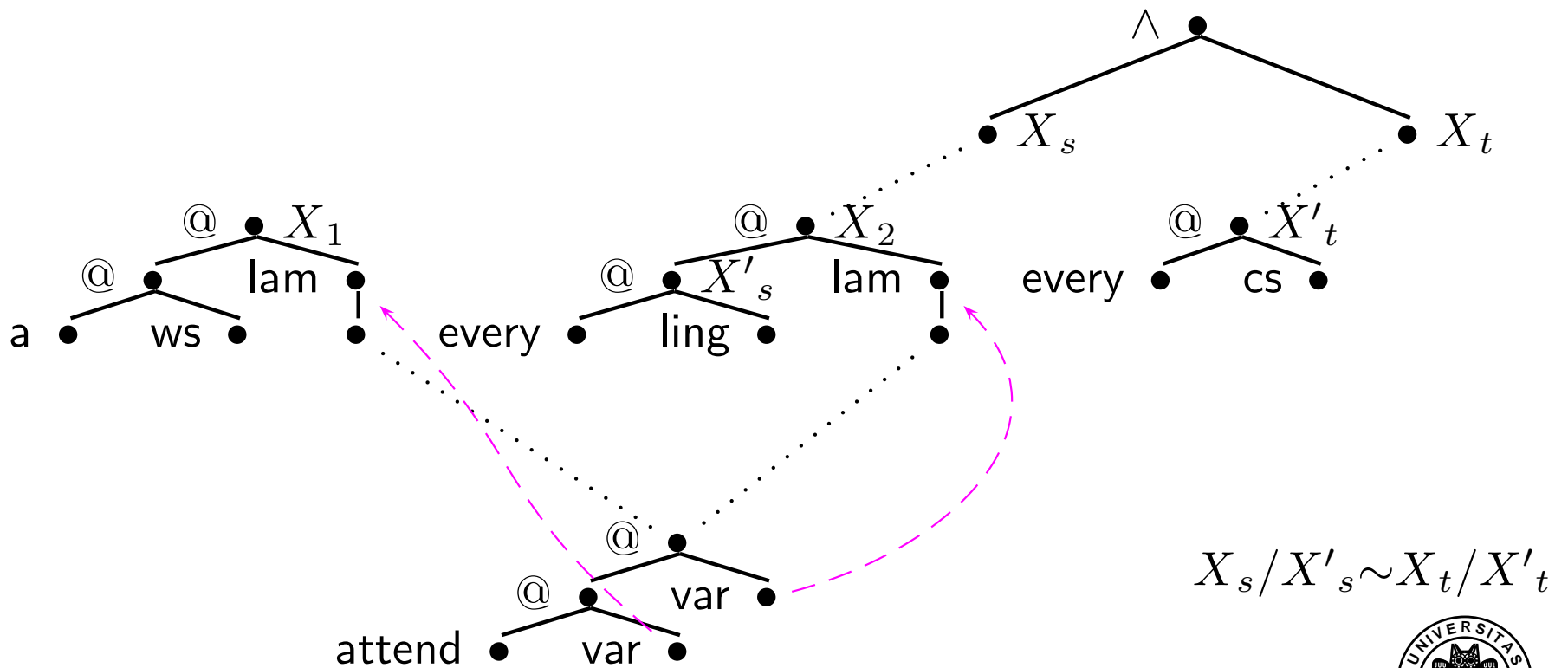
## Scope ambiguity: quantifiers in ellipsis 1

- the source sentence (SS) of (62) [= (54)] has **two** readings
  - (62) Every linguist attended a workshop. Every computer scientist did, too
- its target sentence (TS) has **two** readings, too
- but (54) as a whole has only **three** readings
  - one workshop for all linguists and computer scientists
  - one workshop for all linguists and one (possibly different) workshop for all computer scientists
  - possibly different workshops for all linguists and computer scientists
- the constraint is that the **scope relation must be the same** in both SS and TS
- but at the same time, this relation **cannot be determined yet**



## Scope ambiguity: quantifiers in ellipsis 2

- CLLS strategy with parallelism constraints gets this for free (63)



## Scope ambiguity: quantifiers in ellipsis 3

- the parallelism constraint states that both SS and TS structure must be identical - whatever they might be
- i.e., the scope relation is still open, but must be resolved in the same way
- the indefinite quantifier fragment below  $X_1$  may take scope over the conjunction fragment → first reading
- otherwise, it may take scope above or below the 'universal' fragment below  $X_2$
- since the parallelism constraint  $X_s/X'_s \sim X_t/X'_t$  enforces identity of the constraints below  $X_s$  and  $X_t$ , the scope relation from the SS is bound to reappear in the TS
- but this does not fix the scope relation in advance → two more readings
- similar strategy: Crouch (1995)



## Scope ambiguity: anaphoric dependencies between quantifiers 1

- the problem: an embedded quantifying NP refers anaphorically to its embedding quantifying NP

(64) Every man with a picture of himself has arrived

(65) Every man that I know a child of has arrived

- underlying representation for (64):

$\mathbf{arrive}'(\langle \forall m.\mathbf{man}'(m) \wedge \mathbf{with}'(m, \langle \exists p.\mathbf{picture-of}'(p, m) \rangle) \rangle)$

- after applying the (unchanged) outer term to the predicate it is an argument of:

$\forall m.\mathbf{man}'(m) \wedge \mathbf{with}'(m, \langle \exists p.\mathbf{picture-of}'(p, m) \rangle) \rightarrow \mathbf{arrive}'(m)$



## Scope ambiguity: anaphoric dependencies between quantifiers 2

- giving the remaining term widest scope would lift  $m$  out of the binding domain of the universal quantifier:

$$\exists p. \mathbf{picture-of}'(p, m) \wedge \forall m. \mathbf{man}'(m) \wedge \mathbf{with}'(m, p) \rightarrow \mathbf{arrive}'(m)$$

- the intended representation exhibits narrow scope of the existential quantifier
- it is derived by first discharging the embedded term within the argument of *arrive*

$$\forall m. (\mathbf{man}'(m) \wedge \exists p. (\mathbf{picture-of}'(p, m) \wedge \mathbf{with}'(m, p))) \rightarrow \mathbf{arrive}'(m)$$

- Hobbs and Shieber (1987) restrict applicability of a term to the predicate it is an argument of:

all variable occurrences that are **free in the term** must be **free in the predicate**, too



## Scope ambiguity: anaphoric dependencies between quantifiers 3

- this rules out applying the term in the expression that resulted from applying the unchanged argument of *smile*:

$\forall m.\mathbf{man}'(m) \wedge \mathbf{with}'(m, \langle \exists p.\mathbf{picture-of}'(p, m) \rangle) \rightarrow \mathbf{arrive}'(m)$

- $m$  is free within the term
- but in the predicate (of which the term is an argument) it is bound
- an additional complication: mutual anaphoric dependencies (Bach-Peters sentences):

(66) [Every pilot who shot at it<sub>j</sub>]<sub>i</sub> hit [some MiG that chased him<sub>i</sub>]<sub>j</sub>



# Syntax-semantics interface 1

- in this part of the course, we will focus on two challenges for any syntax-semantics interface:
  - partial modification
    - \* syntactic and semantic structure do not match
    - \* modifiers pertain semantically to only **part** of the expression which they modify syntactically
  - semantic construction for elliptical expressions
    - \* how can one identify source and target sentence?
    - \* how can one identify parallel elements in SS and TS?





## Partial modification: the data 2

- modification of quantifying pronouns: relation to the **restriction** of the quantification introduced in the modified expression

(69) *something blue* 'the set of properties that a blue thing has'

(70) a.  $\lambda P \exists x. \underline{\mathbf{thing}}'(x) \wedge P(x)$

b.  $\lambda P \exists x. \mathbf{thing}'(x) \wedge \mathbf{blue}'(x) \wedge P(x)$

– this expression has only **one** reading



## Partial modification: the data 3

- modification of modifiers: getting the scope right (Kasper to appear)

(71) *potentially controversial plan* ‘maybe not controversial but surely a plan’

(72) *potentially controversial* ‘the intersection of properties  $P$  with the property of being potentially controversial’

- *potentially* may not scope over  $P$ , which instantiates partial modification:
  - **attributive** meaning of *controversial* (**predicative** meaning underlined):

(73)  $\lambda P \lambda x. \underline{\text{controversial}}'(x) \wedge P(x)$

- meaning of *potentially* (‘ $\diamond p$ ’ is true iff  $p$  is true in some possible world):

(74)  $\lambda P \lambda x. \diamond (\wedge P(x))$

- meaning of *potentially controversial*:

(75)  $\lambda P \lambda x. \diamond (\wedge \text{controversial}'(x)) \wedge P(x)$



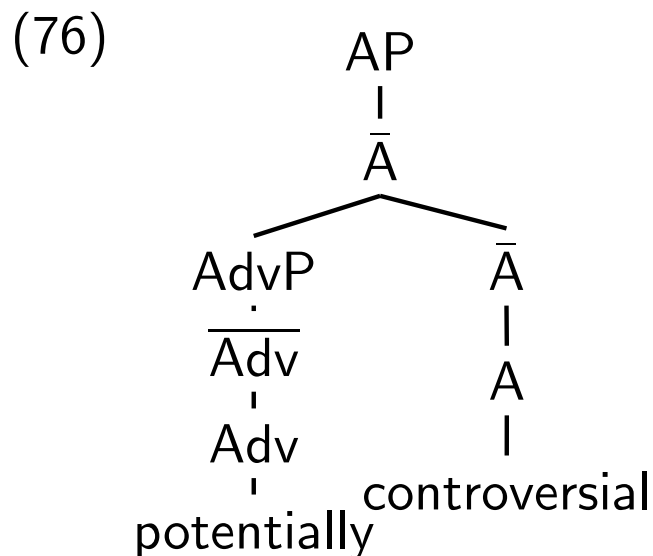
## Partial modification: analysis 1

- the semantics of lexemes (and complex constituents) is described in two fragments, a **main** and an (embedded) **secondary** fragment
- rules of the syntax-semantics interface handle **both kinds** of fragments
- the syntactic structure is **surface-oriented**
- basic approach to the interface mismatches:
  - just like (1), they are described as **scope ambiguities**
  - modifiers dominate the **secondary** fragment of the modified expression, which models its relation to only a part of the meaning of this expression
  - the scope of modifier and **main** fragment of the modified expression is undetermined, which represents potential ambiguities
- i.e., no new machinery is introduced into the interface



## Partial modification: analysis 2

- example: representation and construction of *potentially controversial*



- lexical entry for *controversial*

(77)  $[[A]] : \lambda P \lambda x. \Box(x) \wedge P(x)$

⋮

$[[A_S]] : \text{controversial}'$



## Partial modification: analysis 3

- its semantic representation:

(78)

$$\begin{array}{ccc}
 & \llbracket \text{AP} \rrbracket : \square & \\
 & \vdots & \\
 \llbracket \text{AP}_S \rrbracket : \lambda P \lambda x. \square(x) \wedge P(x) & & \lambda x. \diamond (\wedge \square(x)) \\
 & \text{controversial}' & 
 \end{array}$$

- the potential scope ambiguity between the right and the left fragment is resolved immediately by the types of the fragments
- thus, the sole solution is

(79) [= (75)]  $\lambda P \lambda x. \diamond (\wedge \text{controversial}'(x)) \wedge P(x)$



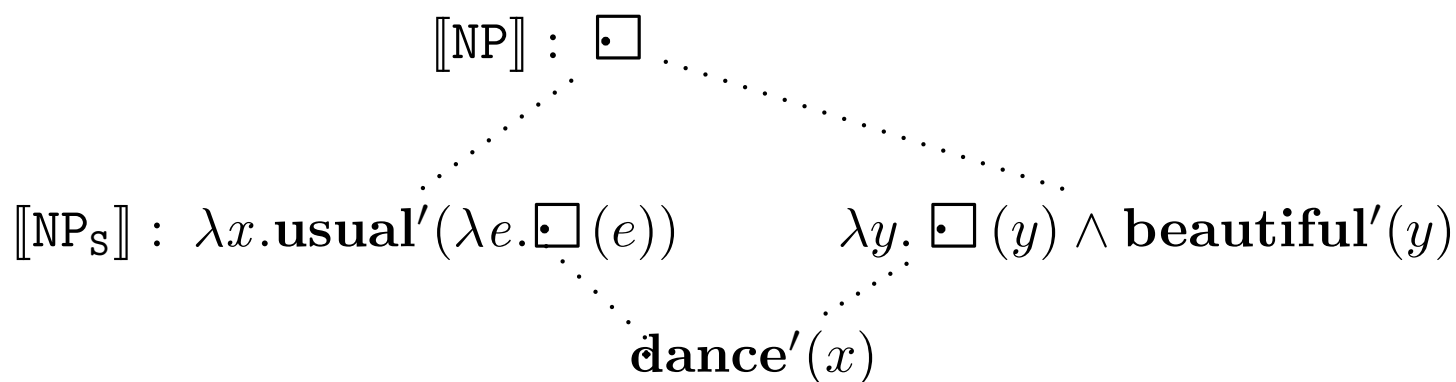
## Partial modification: analysis 4

- the semantics of *beautiful dancer* is derived analogously
- lexical entry for the semantics of *dancer*:

$\llbracket \text{N} \rrbracket : \lambda x.\text{usual}'(\lambda e.\square(e))$

$\llbracket \text{N}_S \rrbracket : \text{dance}'(x)$

- the resulting dominance diamond has **two solutions**, viz., (68b) and (68c)



## Partial modification: further applications

- modification of **change-of-state verbs**: relation to the **aftermath** proposition in the verb semantics (e.g., ‘restitutive’ *again*; Dowty 1979; von Stechow 1996)

(80) Max opened the window again ‘...and it had been open before’

- Müller’s (2003) ‘bracketing paradox’: in nominalisations of German separable-prefix verbs like *Losgerenne*, the morphological structure differs from the scope of the affixes
- Turkish indefinite DPs: modifiers modify  $\bar{D}$  (e.g., *bir kız*), but refer to its NP part semantically

(81) güzel bir kız (‘a beautiful girl’; lit., ‘beautiful a girl’)



## VP ellipsis: introduction 1

- most approaches to the resolution of VP ellipsis presuppose information on the structure of elliptical expressions
- comparatively few approaches (Lappin and McCord 1990; Gregory and Lappin 1998; Lappin 1999; Hardt 1997) deal with the derivation of this information
- Egg and Erk (2002) present an approach that derives this information during semantic construction



## VP ellipsis: introduction 2

(82) John wants to read *Jane Eyre*, and Bill does too

- VP ellipses consist of **source sentence (SS)** and **target sentence (TS)**
- in the TS the VP is a pro-form, e.g., *does too*; the SS-VP is its **antecedent**
- SS and TS have the same meaning except for the semantic contributions of the respective subject NPs
- the existence of several potential SSs gives rise to ambiguity

(83) John wants Max to read everything that Bill does

- TS interpretation 1: ‘Bill reads’
- TS interpretation 2: ‘Bill wants Max to read’
- antecedent-contained deletion in (83): pro-form as part of antecedent



## VP ellipsis: semantic construction 1

- specific syntactic constituent structures are associated with **ellipsis potential**
- ellipsis potential: information on where SS candidates are if VP ellipsis occurs
- examples: sentence conjunction; VPs that consist of a transitive verb and an NP (for ACD)
- ellipsis pro-forms '**discharge**' this potential (by relating TS and SS semantics)
- whole sentences (not just VPs) must be taken into account because of Hirschbühler (1982) examples:

(84) Every linguist attended a workshop. Every computer scientist did, too.



## VP ellipsis: semantic construction 2

- each syntactic constituent carries a  $SEM[antics]$  feature, whose core is a CLLS constraint
- for semantic construction, each constituent distinguishes two node variables by auxiliary features in the  $SEM$  value
  - the value of  $SEM|SCOPE$  models the local scope domain
  - the value of  $SEM|ROOT$  is the uppermost node variable of the semantic contribution of the constituent
- for ellipsis, there are two more auxiliary features in  $SEM$ 
  - $SEM|SUBJ$  is the root of the constraint for the subject of a sentence
  - $SEM|SOS$  is a list of  $SS$  candidates (characterized by tuples of their  $SCOPE$  and  $SUBJ$  values)



## VP ellipsis: semantic construction 3

(85) (a) lex. entry *do (too)*

(b) NP VP  $\rightarrow$  S

$$\dots \mid \text{SEM} \left[ \begin{array}{l} \text{SCOPE} \quad \boxed{1} \\ \text{SOS} \quad \langle \dots \boxed{2} \dots \rangle \\ \text{SUBJ} \quad \boxed{3} \\ \text{CON} \quad s(\boxed{2})/f(\boxed{2}) \sim \boxed{1}/\boxed{3} \end{array} \right]$$

$$\left[ \begin{array}{l} \text{DTR1} \mid \dots \mid \text{SEM} \left[ \begin{array}{l} \text{ROOT} \quad \boxed{1} \\ \text{SCOPE} \quad \boxed{3} \end{array} \right] \\ \text{DTR2} \mid \dots \mid \text{SEM} \left[ \begin{array}{l} \text{SOS} \quad \boxed{2} \\ \text{SUBJ} \quad \boxed{1} \\ \text{SCOPE} \quad \boxed{3} \end{array} \right] \\ \dots \mid \text{SEM} \left[ \begin{array}{l} \text{SOS} \quad \boxed{2} \\ \text{SUBJ} \quad \boxed{1} \\ \text{SCOPE} \quad \boxed{3} \end{array} \right] \end{array} \right]$$

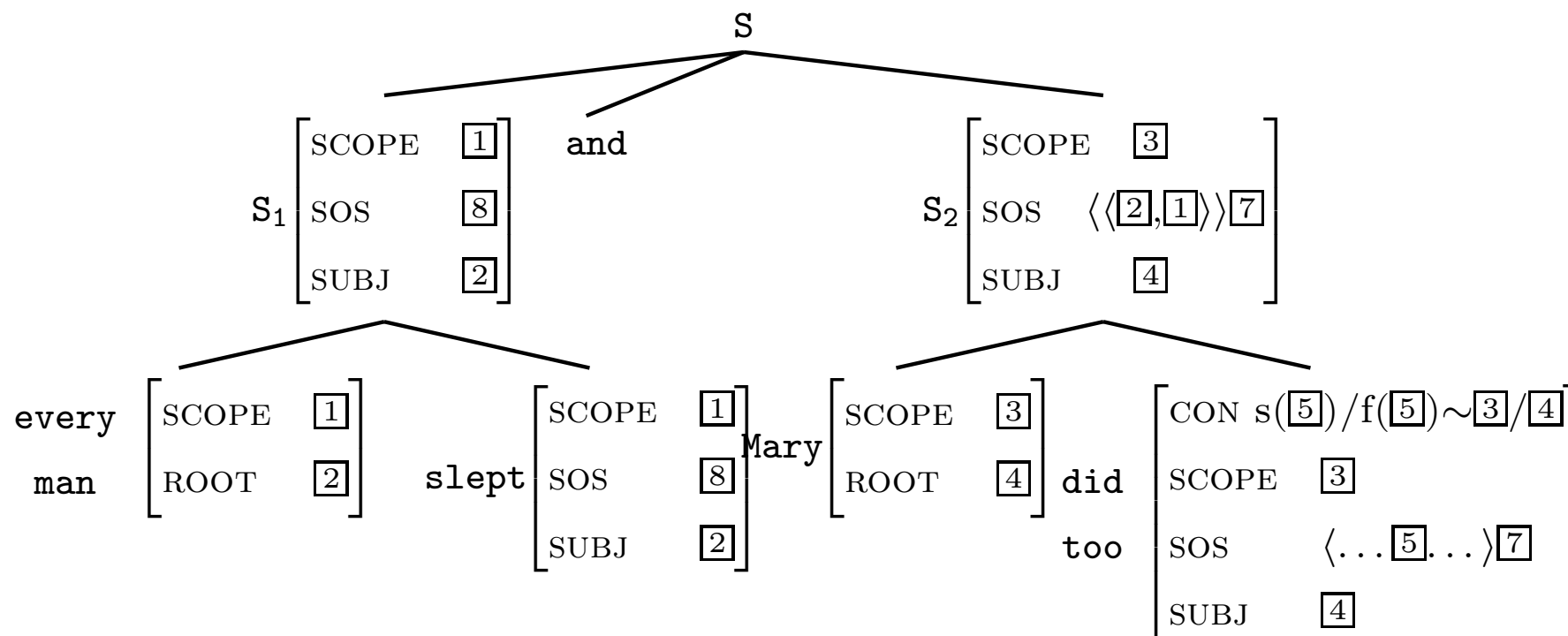
(c) S Conj S  $\rightarrow$  S

$$\left[ \begin{array}{l} \text{DTR1} \mid \dots \mid \text{SEM} \left[ \begin{array}{l} \text{SCOPE} \quad \boxed{1} \\ \text{SUBJ} \quad \boxed{2} \end{array} \right] \\ \text{DTR3} \mid \dots \mid \text{SEM} \mid \text{SOS} \quad \langle \langle \boxed{2}, \boxed{1} \rangle \rangle \end{array} \right]$$



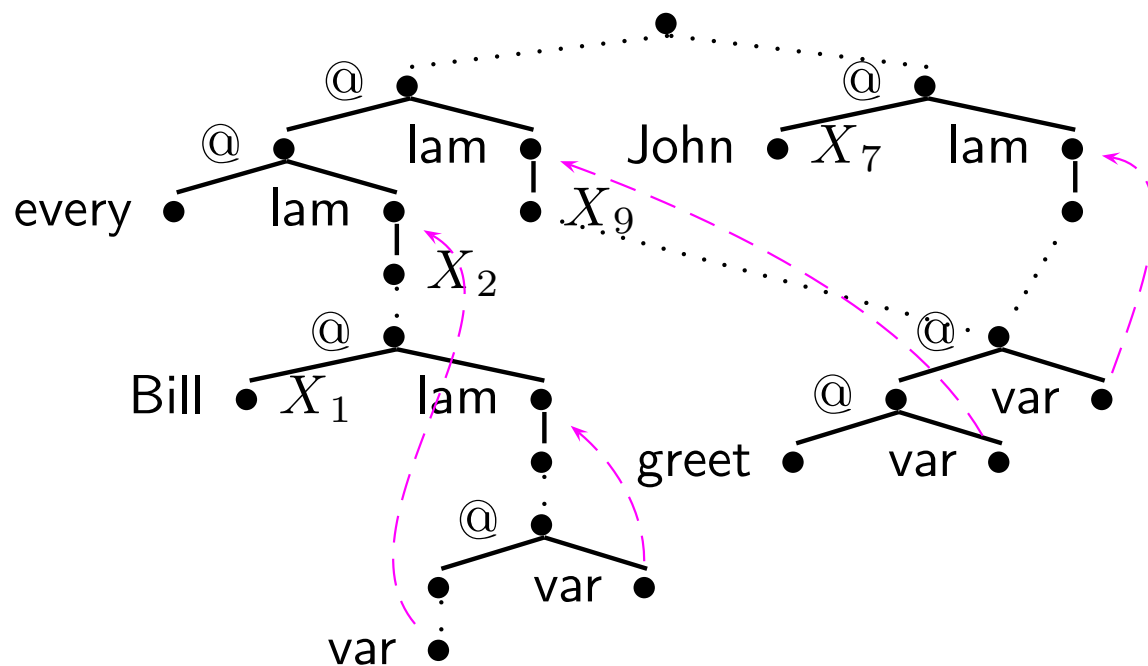
# VP ellipsis: semantic construction 4

(86)



## VP ellipsis: antecedent-contained deletion

*John greeted everyone that Bill did*



$X_9/X_7 \sim X_2/X_1$