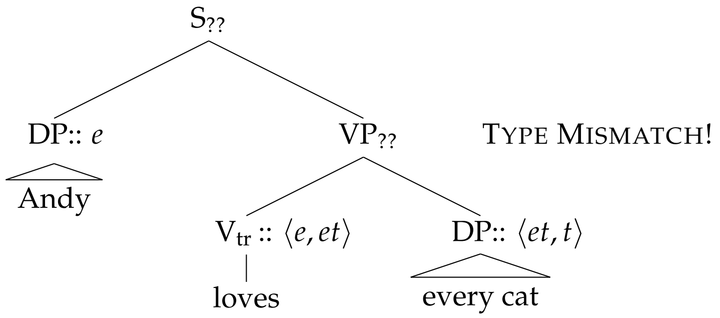
Semantics I, Rutgers University — Week 7
Yimei Xiang — October 17, 2018

Ways of scoping

1. The type-mismatch problem of non-subject GQ

- **Puzzle:** A **type-mismatch** arises when a generalized quantifier appears at a non-subject position.

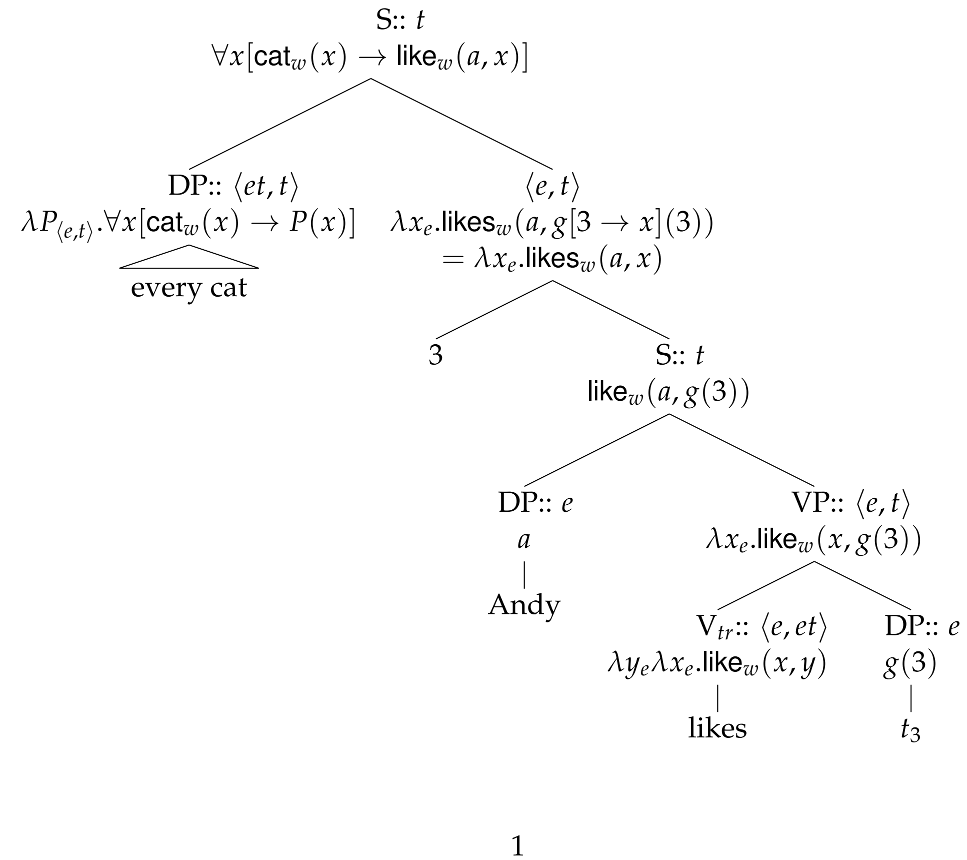
(1) Andy loves every cat.


$S_{??}$
DP:: e
Andy
$VP_{??}$
TYPE MISMATCH!
V_{tr} :: $\langle e, et\rangle$
loves
DP:: $\langle et, t\rangle$
every cat

1.1. Solving type-mismatch via QR

- This type-mismatch can be resolved by a covert LF movement of the generalized quantifier, called **Quantifier Raising** (QR). In syntax, the generalized quantifier *every cat* is moved to the left edge of the sentence, leaving a trace. In semantics, we interpret this trace as a variable of a matching type (type *e* here), and then abstract over this trace and create a predicate via Predicate Abstraction. The moved generalized quantifier combines with this predicate via Functional Application.

(2)


S:: t
$\forall x[\mathsf{cat}_w(x) \rightarrow \mathsf{like}_w(a, x)]$
DP:: $\langle et, t\rangle$
$\lambda P_{\langle e,t\rangle}.\forall x[\mathsf{cat}_w(x) \rightarrow P(x)]$
every cat
$\langle e, t\rangle$
$\lambda x_e.\mathsf{likes}_w(a, g[3 \rightarrow x](3))$
$= \lambda x_e.\mathsf{likes}_w(a, x)$
3
S:: t
$\mathsf{like}_w(a, g(3))$
DP:: e
a
Andy
VP:: $\langle e, t\rangle$
$\lambda x_e.\mathsf{like}_w(x, g(3))$
V_{tr}:: $\langle e, et\rangle$
$\lambda y_e \lambda x_e.\mathsf{like}_w(x, y)$
likes
DP:: e
$g(3)$
t_3

- **If a generalized quantifier occurs at a subject position, does it move at all?**

  The modern syntactic theory says "Yes". But this movement is not driven by type-mismatch, but by syntactic reasons – VP-subject hypothesis. It occurs at the transformation from DS to SS.

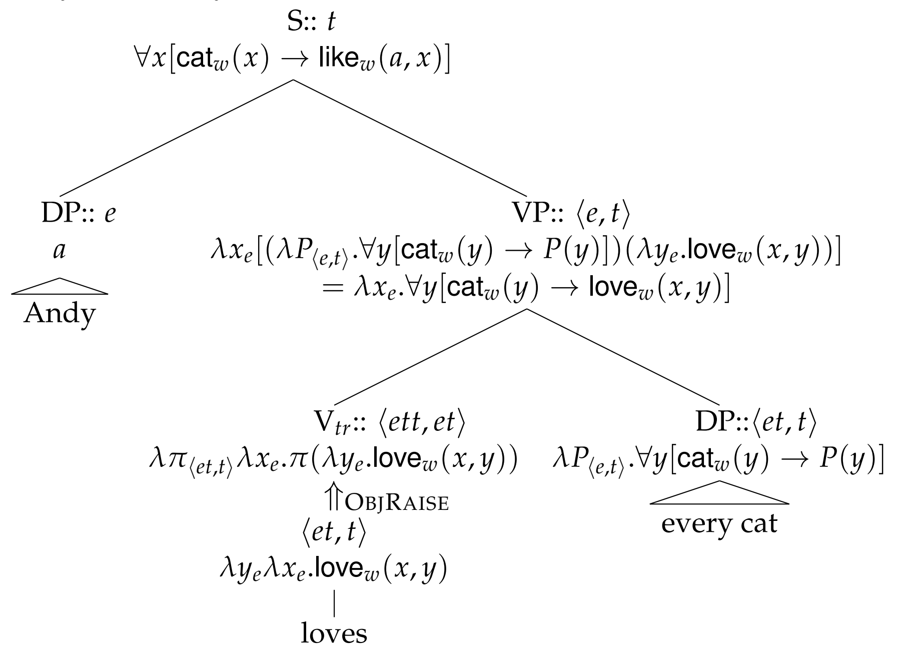
1.2. Solving type-mismatch via type-shifting

- Another approach is to solve the type-mismatch problem is to interpret the quantifier phrase in situ but apply a type-shifting operation to change either the type of the quantifier phrase or the type of the verb.
- Representative type-shifting operations include *argument raising* for verbs (Hendriks 1993), *function-argument flip flop* (Partee and Rooth 1983), and the continuation passing style (CPS) transforms used in continuation-based works (Shan and Barker 2006, among others).
- **Option 1**: Type-shifting the verb
  We can lift the type of the object argument of the verb, so that the verb can be applied to a generalized quantifier.

  (3) **Object Type-Raising Rule** (Hendriks 1993)
  If R is of type $\langle e, et\rangle$, then $\Uparrow_{\textsc{ObjRaise}}(R) = \lambda\pi_{\langle et,t\rangle}\lambda x_e.\pi(\lambda y_e.R(x,y))$

  For example:

  (4) Andy loves every cat.


S:: t
$\forall x[\mathsf{cat}_w(x) \rightarrow \mathsf{like}_w(a,x)]$
DP:: e
a
Andy
VP:: $\langle e,t\rangle$
$\lambda x_e[(\lambda P_{\langle e,t\rangle}.\forall y[\mathsf{cat}_w(y) \rightarrow P(y)])(\lambda y_e.\mathsf{love}_w(x,y))]$
$= \lambda x_e.\forall y[\mathsf{cat}_w(y) \rightarrow \mathsf{love}_w(x,y)]$
V$_{tr}$:: $\langle ett, et\rangle$
$\lambda\pi_{\langle et,t\rangle}\lambda x_e.\pi(\lambda y_e.\mathsf{love}_w(x,y))$
$\Uparrow_{\textsc{ObjRaise}}$
$\langle et,t\rangle$
$\lambda y_e\lambda x_e.\mathsf{love}_w(x,y)$
loves
DP::$\langle et,t\rangle$
$\lambda P_{\langle e,t\rangle}.\forall y[\mathsf{cat}_w(y) \rightarrow P(y)]$
every cat

- **Option 2**: Type-shifting the non-subject quantifier
  We can change the type of the generalized quantifier so that it can select for a transitive verb.

  (5) **Quantifier Type-Raising Rule**
  If π is of type $\langle et,t\rangle$, then $\Uparrow_{\textsc{QRaise}}(\pi) = \lambda R_{\langle e,et\rangle}\lambda x_e.\pi(\lambda y_e.R(x,y))$

  For example:

  (6) a. $\Uparrow_{\textsc{QRaise}}([\![\text{every cat}]\!]^w) = \lambda R_{\langle e,et\rangle}\lambda x_e.\forall y[\mathsf{cat}_w(y) \rightarrow R(x,y)]$
  b. $(\Uparrow_{\textsc{QRaise}}([\![\text{every cat}]\!]^w))([\![\text{love}]\!]^w) = \lambda x_e.\forall y[\mathsf{cat}_w(y) \rightarrow \mathsf{love}_w(x,y)]$

Discussion: Assume the following structure for a sentence with a ditransitive verb *show*.

(7) Andy showed Billy Cindy.
$[_S$ Andy $[_{VP}$ $[_{VP}$ showed Billy] Cindy]]

Now, consider the following sentence. Can we solve the type-mismatch between *showed* and *nobody* by the above type-shifting rules? Can we solve it by QR?

(8) Andy showed nobody Cindy.

2. Scope ambiguity

- In many cases, sentences with multiple scopal expressions are ambiguous between a **surface scope reading** (consistent with the surface linear order of the scopal expressions in the sentence) and an **inverse scope** reading (where the order of interpretation of the scopal expressions is the inverse of linear order).

(9) Andy did**n't** read **a book**.
  a. Andy didn't read any books. *not ≫ a book*
  b. There is a book that Andy didn't read. *a book ≫ not*

(10) **Every kid** loves **a cat**.
  a. For every kid there is a cat that s/he loves. *every kid ≫ a cat*
  b. There is a cat loved by every kid. *a cat ≫ every kid*

(11) **Every student of mine** did**n't** show up on time.
  a. None of the students showed up on time. *every ≫ not*
  b. It is not the case that every student of mine showed up on time. *not ≫ every*

- NB: not every sentence with multiple scopal elements admits an inverse scope reading.

– Case 1: *Modal and negation*

(12) a. John is not allowed to read this book.
  b. John is allowed not to read this book.

Negation and modal verbs do not undertake covert scope shifting operations.

– Case 2: Syntax constraints

(13) a. John met everyone who admires three midwestern cities.
  b. John met someone who inhabits three midwestern cities.

Bare numeral QNPs do not take inverse wide scope out of islands.

– Case 3: *Quantifiers and negation* (Beghelli and Stowell 1997)

(14) a. John didn't meet every student of mine on time.
  b. A student of mine didn't show up on time.

Generalized Scope Economy condition (Mayr & Spector 2012, cf. Fox 1995, 2000): A covert scope shifting operation cannot apply if the meaning of the resulting reading is equivalent to or stronger than (i.e. entails) the meaning that would have resulted without it.

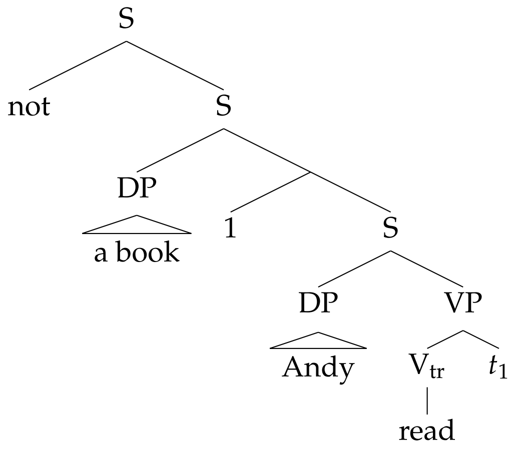
2.1. Scoping via quantifier raising

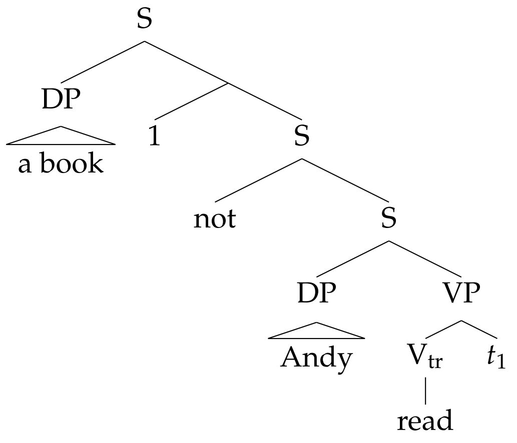
- A generalized quantifier can undertake QR and adjoin to different sentential nodes. Which sentential node it adjoins to determines its semantic scope.

(15) Andy didn't read a book.

a. $\neg\exists x[\mathsf{book}_w(x) \wedge \mathsf{read}_w(a, x)]$

b. $\exists x[\mathsf{book}_w(x) \wedge \neg\mathsf{read}_w(a, x)]$

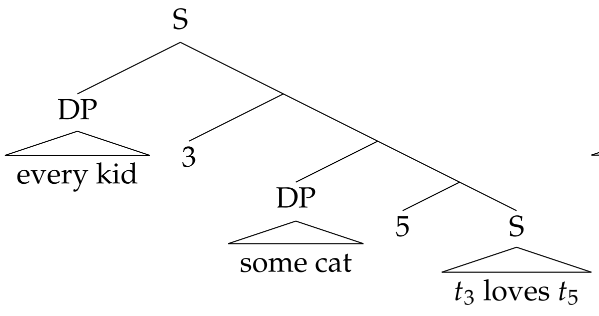


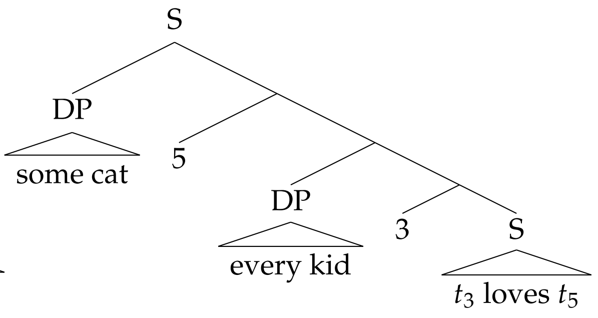


(16) Every kid loves a cat.

a. $\forall x[\mathsf{kid}_w(x) \to \exists y[\mathsf{cat}_w(y) \wedge \mathsf{love}_w(x, y)]]$

b. $\exists y[\mathsf{cat}_w(x) \wedge \forall x[\mathsf{kid}_w(x) \to \mathsf{love}_w(x, y)]]$





Exercise: Compose the trees in (16) node by node.

Exercise: For each of the following sentences, draw a tree diagram to illustrate its LF.

(17) a. One apple in every basket is rotten.

b. I require no student to visit my office hour.

2.2. Semantic reconstruction

- The puzzle: Recall that negation doesn't move. How can we derive the inverse scope reading of the following sentence?

(18) **Every student** did**n't** stay.

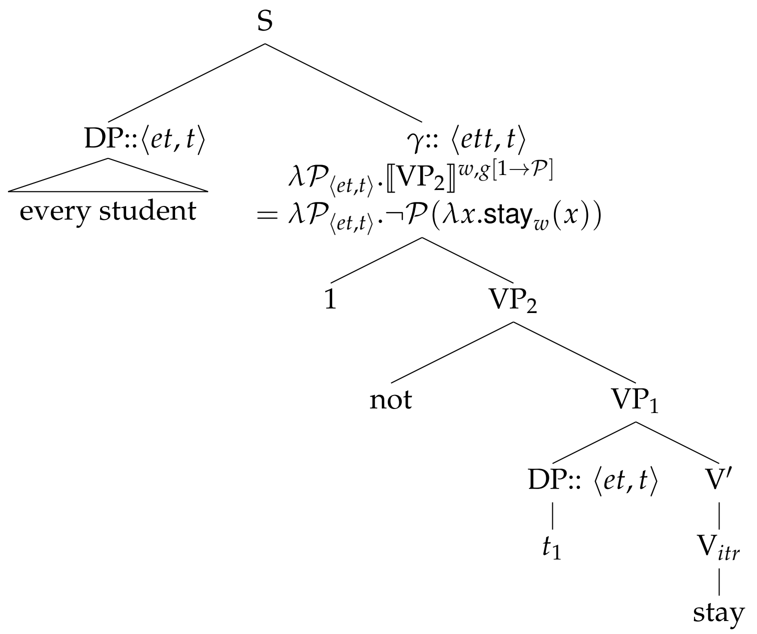
a. None of the students left. *every ≫ not*

b. It is not the case that every student stayed. *not ≫ every*

Adopting VP-subject hypothesis, we can assume that the VP-internal trace of *every student* is of type $\langle et, t\rangle$ (Use the Trace Interpreting Rule: the assignment function g maps an index 1 to a value of type $\langle et, t\rangle$.) In cases as such, Predicate Abstraction can introduce functions from generalized quantifiers into values.

- **Option 1: One single movement**

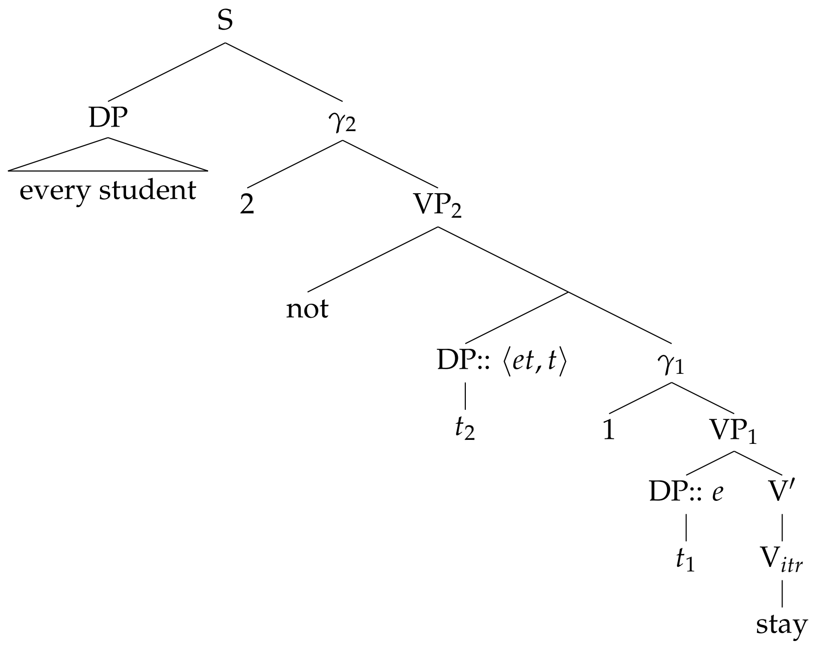
(19)


S
DP::$\langle et, t\rangle$
every student
γ:: $\langle ett, t\rangle$
$\lambda\mathcal{P}_{\langle et,t\rangle}.[\![\text{VP}_2]\!]^{w,g[1\to\mathcal{P}]}$
$= \lambda\mathcal{P}_{\langle et,t\rangle}.\neg\mathcal{P}(\lambda x.\mathsf{stay}_w(x))$
1
VP_2
not
VP_1
DP:: $\langle et, t\rangle$
t_1
V′
V_{itr}
stay

- **Option 2: Two movements**

  The generalized quantifier firstly takes a short movement to a position under negation, leaving a trace of type e, and then it moves again, leaving a trace of type $\langle et, t\rangle$.

(20)



- Semantic reconstruction is commonly used for explaining **A-movement**.

(21) a. Someone is likely to arrive.
b. A semanticist might be at the conference.

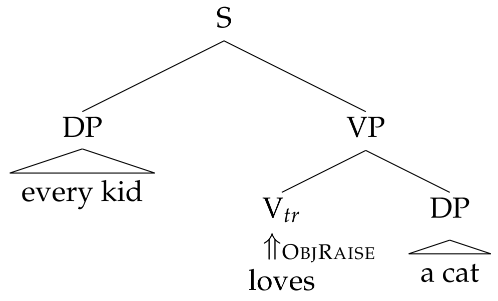
Discussion: How can we derive the *not* $\gg$ *every phil* $\gg$ *a ling* reading for the following sentence? Is it possible to derive this reading using Option 1 in (19)?

(22) Every philosopher didn't invite a linguist.
'It is not the case that every philosopher invited a linguist.' (*not* $\gg$ *every phil* $\gg$ *a ling*)

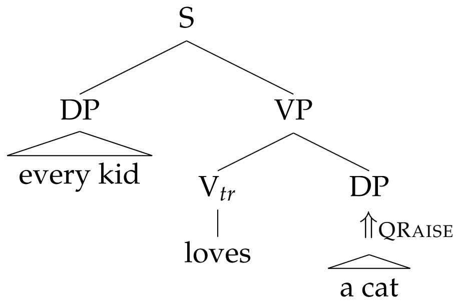
2.3. Scoping via type-shifting

- Discussion: Do the following structures yield a surface scope reading or an inverse scope reading?

(23) a.



b.



- **Option 1**: Type-shifting the verb (Hendriks 1993, read Coppock & Champollion 2018: pp. 226-227)
We can raise the types of both the object argument and the subject argument of the verb, so that this transitive verb combines with two generalized quantifiers. If we raise the object first, we get the surface scope reading; if we raise the subject first, we get the inverse scope reading.

- **Option 2**: (Internally) Type-lifting the generalized quantifiers
The continuation-based grammar (Barker 2002, among others) allows type-lifting to be applied externally or internally. Internally lifting an expression to a higher type makes it take wider scope, including scoping over elements that precede it.

(24) $m = \lambda k.m(\lambda v.k(v))$
a. External lift
$m^{\uparrow} = \lambda Q.Q(\lambda k.m(\lambda v.k(v)))$
b. Internal lift
$m^{\uparrow\uparrow} = \lambda Q.m(\lambda v.Q(\lambda k.k(v)))$