LING5702: Lecture Notes 19
Quantifier Scope

Sentences can often have multiple different readings, depending on **scope**:

- *Two people are in each booth.*

These sentences can be understood with an **in-situ** interpretation:

$$\begin{array}{l}
\textsf{Two}\ (\lambda_x\ \textsf{Person}\ x) \\
\qquad (\lambda_x\ \textsf{All}\ (\lambda_y\ \textsf{Booth}\ y) \\
\qquad\qquad (\lambda_y\ \textsf{Some}\ (\lambda_e\ \textsf{In}\ y\ x\ e) \\
\qquad\qquad\qquad\qquad (\lambda_e\ \textsf{True})))
\end{array}$$

or a **raised** interpretation:

$$\begin{array}{l}
\textsf{All}\ (\lambda_y\ \textsf{Booth}\ y) \\
\qquad (\lambda_y\ \textsf{Two}\ (\lambda_x\ \textsf{Person}\ x) \\
\qquad\qquad (\lambda_x\ \textsf{Some}\ (\lambda_e\ \textsf{In}\ y\ x\ e) \\
\qquad\qquad\qquad\qquad (\lambda_e\ \textsf{True})))
\end{array}$$

The raised interpretation is often non-local (can cross clause boundaries):

- *At least two CEOs said a representative was going to each country.*

Contents

19.1 Evidence for explicit scoping [Dotlačil & Brasoveanu, 2015]

Scope seems to be explicitly calculated after each sentence (i.e. doesn't remain underspecified):

- **stimuli:** sentences presented in eye-tracking:
  (a) *A caregiver comforted a child every night. The caregivers wanted the <u>children</u> to...*
  (b) *A caregiver comforted a child every night. The caregivers wanted the <u>child</u> to...*
  (c) *A caregiver comforted a child every night. The caregiver wanted the <u>children</u> to...*
  (d) *A caregiver comforted a child every night. The caregiver wanted the <u>child</u> to...*

  These analyses are eliminated at *caregiver*, but neither is the preferred in-situ analysis:

$$\begin{array}{l}
\textsf{All}\ (\lambda_t\ \textsf{Night}\ t) \\
\quad (\lambda_t\ \textsf{Some}\ (\lambda_k\ \textsf{Caregiver}\ k) \\
\qquad\quad (\lambda_k\ \textsf{Some}\ (\lambda_c\ \textsf{Child}\ c) \\
\qquad\qquad\quad (\lambda_c\ \textsf{Comfort}\ t\ k\ c)))
\end{array}
\qquad
\begin{array}{l}
\textsf{Some}\ (\lambda_c\ \textsf{Child}\ c) \\
\quad (\lambda_c\ \textsf{All}\ (\lambda_t\ \textsf{Night}\ t) \\
\qquad\quad (\lambda_t\ \textsf{Some}\ (\lambda_k\ \textsf{Caregiver}\ k) \\
\qquad\qquad\quad (\lambda_k\ \textsf{Comfort}\ t\ k\ c)))
\end{array}$$

The preferred in-situ (first) analysis is eliminated at *children*:

$$\begin{array}{l} \textsf{Some}\ (\lambda_k\ \textsf{Caregiver}\ k) \\ \quad (\lambda_k\ \textsf{Some}\ (\lambda_c\ \textsf{Child}\ c) \\ \qquad (\lambda_c\ \textsf{All}\ (\lambda_t\ \textsf{Night}\ t) \\ \qquad\quad (\lambda_t\ \textsf{Comfort}\ t\ k\ c))) \end{array} \qquad \begin{array}{l} \textsf{Some}\ (\lambda_k\ \textsf{Caregiver}\ k) \\ \quad (\lambda_k\ \textsf{All}\ (\lambda_t\ \textsf{Night}\ t) \\ \qquad (\lambda_t\ \textsf{Some}\ (\lambda_c\ \textsf{Child}\ c) \\ \qquad\quad (\lambda_c\ \textsf{Comfort}\ t\ k\ c))) \end{array}$$

- **measure:** eye-tracking fixation durations at *children* (and spillover word).
- **results:** singular-plural (c) is slowest at *children*, suggests dynamic reanalysis there.

19.2 Computational-level scope resolution [Cooper, 1983, Keller, 1988]

We can define scope raising at the computational level, using logic.

First we generalize across types using **schemas**, defined with meta-variables γ and δ.

Specifically, we allow functions to take any type γ_n with any number n of arguments $\delta_1, ..., \delta_n$:

$$\begin{aligned} \gamma_0 &= \mathrm{t} \\ \gamma_n &= \langle \delta_n, \gamma_{n-1} \rangle \end{aligned}$$

Then we augment our semantics with a **store** or **context** Γ, Δ, Θ, delimited by '$\vdash$':

$$\underbrace{\underbrace{\Delta}_{\text{store}} \vdash \underbrace{\varphi : \underbrace{\langle \gamma_n, \gamma_{n-1} \rangle}_{\text{type}}}_{\text{expression}}}_{\text{sequent}}$$

These are called **sequents**. The store Γ, Δ, Θ of each sequent is a list of other sequents and variables.

Quantifier functions φ can now be stored, leaving variables x of type δ_n in their place:

$$\underbrace{\Delta \vdash \varphi : \langle \gamma_n, \gamma_{n-1} \rangle}_{\text{sequent}} \;\Rightarrow\; (\underbrace{\Delta \vdash \varphi : \langle \gamma_n, \gamma_{n-1} \rangle}_{\text{stored sequent}}, \underbrace{x : \delta_n}_{\text{variable}}) \vdash x : \delta_n \qquad \text{(Quantifier Storage)}$$

Stored functions φ are then retrieved and applied to bind the new variable x at a wider scope ψ:

$$\Gamma, (\underbrace{\Delta \vdash \varphi : \langle \gamma_n, \gamma_{n-1} \rangle}_{\text{stored sequent}}, \underbrace{x : \delta_n}_{\text{variable}}), \Theta \vdash \psi : \gamma_{n-1} \;\Rightarrow\; \Gamma, \Delta, \Theta \vdash (\varphi\ (\lambda_{x:\delta_n}\ \psi)) : \gamma_{n-1} \quad \text{(Quantifier Retrieval)}$$

Note the retrieved sequents and functions need not be retrieved in the same order they are stored!

Other rules are then augmented with stores that are just concatenated without being changed.

$$\Gamma \vdash \varphi : \langle \alpha, \beta \rangle \quad \Delta \vdash \chi : \alpha \quad \Rightarrow \quad \Gamma, \Delta \vdash (\varphi\ \chi) : \beta \qquad \text{(Forward Function Application)}$$
$$\Gamma \vdash \chi : \alpha \quad \Delta \vdash \varphi : \langle \alpha, \beta \rangle \quad \Rightarrow \quad \Gamma, \Delta \vdash (\varphi\ \chi) : \beta \qquad \text{(Backward Function Application)}$$

and similarly for Modification and Closure rules.

Now stored sequents can propagate up the translation and be retrieved in any order!

This approach to quantifier scope raising is called **(nested) Cooper storage**.

Now we can translate our preferred reading of *Two people occupy every booth*:

$\vdash$ (All Booth ($\lambda_{y:e}$ Two Person (In y))) : t

($\vdash$ (All Booth) : $\langle\langle$e, t$\rangle$, t$\rangle$, y : e) $\vdash$ (Two Person (In y)) : t

$\vdash$ (Two Person) : $\langle\langle$e, t$\rangle$, t$\rangle$ ($\vdash$ (All Booth) : $\langle\langle$e, t$\rangle$, t$\rangle$, y : e) $\vdash$ (In y) : $\langle$e, t$\rangle$

$\vdash$ Two : $\langle\langle$e, t$\rangle$, $\langle\langle$e, t$\rangle$, t$\rangle\rangle$ $\vdash$ Person : $\langle$e, t$\rangle$ $\vdash$ In : $\langle$e, $\langle$e, t$\rangle\rangle$ ($\vdash$ (All Booth) : $\langle\langle$e, t$\rangle$, t$\rangle$, y : e) $\vdash$ y : e

Two *people* *occupy* $\vdash$ (All Booth) : $\langle\langle$e, t$\rangle$, t$\rangle$

$\vdash$ All : $\langle\langle$e, t$\rangle$, $\langle\langle$e, t$\rangle$, t$\rangle\rangle$ $\vdash$ Booth : $\langle$e, t$\rangle$

every *booth*

We can also translate our in-situ reading without using schematized quantifiers:

(Two Person ($\lambda_{x:e}$ All Booth ($\lambda_{y:e}$ In y x))) : t

($\vdash$ (Two Person) : $\langle\langle$e, t$\rangle$, t$\rangle$, x : e) $\vdash$ (All Booth ($\lambda_{y:e}$ In y x)) : t

($\vdash$ (Two Person) : $\langle\langle$e, t$\rangle$, t$\rangle$, x : e), ($\vdash$ (All Booth) : $\langle\langle$e, t$\rangle$, t$\rangle$, y : e) $\vdash$ (In y x) : t

($\vdash$ (Two Person) : $\langle\langle$e, t$\rangle$, t$\rangle$, x : e) $\vdash$ x : e ($\vdash$ (All Booth) : $\langle\langle$e, t$\rangle$, t$\rangle$, y : e) $\vdash$ (In y) : $\langle$e, t$\rangle$

$\vdash$ (Two Person) : $\langle\langle$e, t$\rangle$, t$\rangle$ $\vdash$ In : $\langle$e, $\langle$e, t$\rangle\rangle$ ($\vdash$ (All Booth) : $\langle\langle$e, t$\rangle$, t$\rangle$, y : e) $\vdash$ y : e

$\vdash$ Two : $\langle\langle$e, t$\rangle$, $\langle\langle$e, t$\rangle$, t$\rangle\rangle$ $\vdash$ Person : $\langle$e, t$\rangle$ *occupy* $\vdash$ (All Booth) : $\langle\langle$e, t$\rangle$, t$\rangle$

Two *people* $\vdash$ All : $\langle\langle$e, t$\rangle$, $\langle\langle$e, t$\rangle$, t$\rangle\rangle$ $\vdash$ Booth : $\langle$e, t$\rangle$

every *booth*

We can likewise model negation without schemas, since (recall) it can be modeled as quantification.

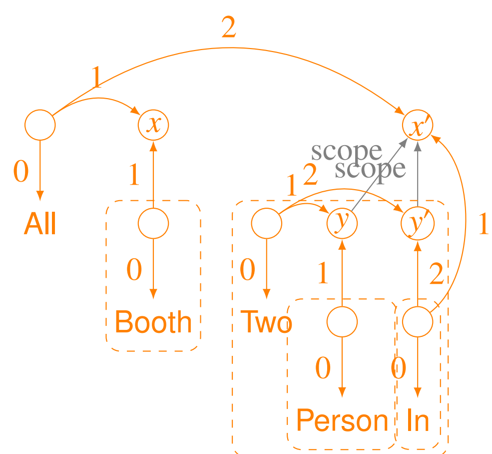
> **Practice 19.1: trees with sequents**
>
> Draw **a translation tree with logical sequents at each branch** for the phrase:
>
> *each country*
>
> in which *each country* undergoes storage.

> **Practice 19.2: trees with sequents**
>
> Draw **a translation tree with logical sequents at each branch** for the following sentence:
>
> *A city in each country is coastal.*
>
> in which *each country* is scoped high.

19.3 Algorithmic-level scope resolution [Schuler & Wheeler, 2014]

At the algorithmic level, scope can be added after processing using extra cued associations:

$$\mathsf{All}\,(\lambda_x \boxed{\mathsf{Booth}\ x})\,(\lambda_{x'} \boxed{\mathsf{Two}\,(\lambda_y \boxed{\mathsf{Person}\ y})\,(\lambda_{y'} \boxed{\mathsf{In}\ x'\ y'})}) \Leftrightarrow$$



We'll assume the following constants (with a localist representation: referential states are δ_v):

1. $V \in \mathbb{R}$: a maximum number of referential states (variables in lambda calculus expressions);
2. $\mathbf{q} \in \{0,1\}^V$: a vector of zeros or ones indicating if each referential state is a quantification;
3. $\mathbf{v} \in \mathbb{R}^V$: a vector of precedence ('readiness') values for each referential state, based on:
    (a) quantifier type (e.g. Each has low precedence, so it usually scopes last/highest)
    (b) participated-in predicates (e.g. y in In x y will scope higher than x)
    (c) order in sentence (this enforces a preference for in-situ scope)

4. $\mathbf{E}_n \in \mathbb{R}^{V \times V}$: a matrix of associations from functions to arguments numbered by n;

We'll also assume **inheritance** associations ('rin') from the lecture notes on sentence processing:

$$\mathbf{E}_{\text{rin}} = \mathbf{E}_1 \, \text{diag}(\mathbf{q}) \, \mathbf{E}_2^{\top}$$

We'll need **closure** matrices directly associating states connected by any number of associations:

$$\mathbf{E}_{\text{P}} = \mathbf{I} + \sum_{n=1}^{N} \prod_{i=1}^{n} \sum_{\ell \in \{1,2,3,\ldots\}} \mathbf{E}_{\ell} \, \text{diag}(\mathbf{1}{-}\mathbf{q}) + \text{diag}(\mathbf{1}{-}\mathbf{q}) \, \mathbf{E}_{\ell}^{\top}$$
$$\mathbf{E}_{\text{I}} = \mathbf{I} + \sum_{n=1}^{N} \prod_{i=1}^{n} \sum_{\ell \in \{\text{cin},\text{ein},\text{rin}\}} \mathbf{E}_{\ell} + \mathbf{E}_{\ell}^{\top}$$

First, initialize iteration-dependent variables:

1. $\mathbf{Q}_0 = \mathbf{0}^{V \times V}$: an initially empty matrix of immediate outscopings;
2. $\mathbf{P}_0 = \mathbf{E}_{\text{P}} + \mathbf{I} - \text{diag}(\mathbf{E}_{\text{P}})$: a matrix of fully-connected partitions, starting with no inheritances;
3. $\mathbf{u}_0 = \sum_{v \text{ s.t. } v = \text{argmax} \, \text{diag}(\mathbf{v}) \, \mathbf{P}_0 \, \delta_v} \delta_v$: a vector of used referential states, starting with the readiest.

Then, for each iteration $i \in \{1, 2, 3, \ldots\}$ such that some states remain un-used ($\mathbf{u}_{i-1} \neq \mathbf{1}$):

1. $u_i = \text{argmax} \, \text{diag}(\mathbf{v}) \, \underbrace{\text{diag}(\mathbf{1}{-}\mathbf{E}_{\text{I}} \, (\mathbf{1}{-}\mathbf{u}_{i-1}))}_{\text{not connected to unused}} (\mathbf{1}{-}\mathbf{Q}_{i-1}{}^{\top} \mathbf{1})$: get readiest used un-scoped state;
2. $\mathbf{P}_i = \mathbf{a} \, \mathbf{a}^{\top} + \underbrace{\mathbf{P}_{i-1} \, \text{diag}(\mathbf{1}{-}\mathbf{a})}_{\text{copy non-merged partitions}}$ where $\mathbf{a} = \mathbf{P}_{i-1} \, \mathbf{E}_{\text{I}} \, \delta_{u_i}$: merge partitions connected via u_i;
3. $v_i = \text{argmax} \, \text{diag}(\mathbf{v}) \, \text{diag}(\mathbf{1}{-}\mathbf{u}_{i-1}) \, \mathbf{P}_i \, \delta_{u_i}$: find readiest unused state in new partition;
4. $\mathbf{Q}_i = \mathbf{Q}_{i-1} + \delta_{v_i} \, \delta_{u_i}{}^{\top} \mathbf{E}_{\text{I}}$: associate referential states in scope matrix;
5. $\mathbf{u}_i = \mathbf{u}_{i-1} + \delta_{v_i}$: add v_i as used.

Participant and scope associations define lambda calculus expressions as described earlier.

References

[Cooper, 1983] Cooper, R. (1983). *Quantification and syntactic theory*. Dordrecht, Holland: D. Reidel.

[Dotlačil & Brasoveanu, 2015] Dotlačil, J. & Brasoveanu, A. (2015). The manner and time course of updating quantifier scope representations in discourse. *Language, Cognition and Neuroscience*, 30(3), 305–323.

[Keller, 1988] Keller, W. R. (1988). Nested cooper storage: The proper treatment of quantifiers in ordinary noun phrases. In E. U. Reyle & E. C. Rohrer (Eds.), *Natural Language Parsing and Linguistic Theories* (pp. 432–447). D. Reidel.

[Schuler & Wheeler, 2014] Schuler, W. & Wheeler, A. (2014). Cognitive compositional semantics using continuation dependencies. In *Third Joint Conference on Lexical and Computational Semantics (*SEM'14)*.