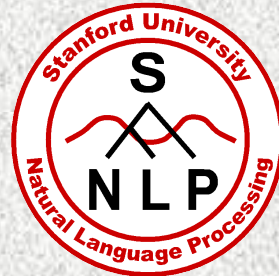# Scope ambiguity and underspecified representations

Bill MacCartney

CS224U: Natural Language Understanding

14 February 2012

[some slides adapted from David Ahn and Alexander Koller]

Quantifier scope ambiguity

In this country, a woman gives birth every 15 minutes.
Our job is to find that woman and stop her.

— Groucho Marx celebrates quantifier scope ambiguity

What are the possible readings here?

$\forall f$ (fifteen-minutes(f) $\rightarrow$ $\exists w$ (woman(w) $\land$ gives-birth-during(w, f)))

$\exists w$ (woman(w) $\land$ $\forall f$ (fifteen-minutes(f) $\rightarrow$ gives-birth-during(w, f)))

Surprisingly, both readings are available in English!

Which one is the joke meaning?

If you liked that one …

A cyclist is hit by a tram every day in Amsterdam.
He doesn't appreciate it very much.

What are the possible readings here?

$\forall d\ (day(d) \rightarrow \exists c\ (cyclist(c) \wedge \text{hit-by-tram-on-day}(c, d)))$

$\exists c\ (cyclist(c) \wedge \forall d\ (day(f) \rightarrow \text{hit-by-tram-on-day}(c, d)))$

How does this joke work?

Sometimes, context can resolve scope ambiguities.

Where scope ambiguity matters!

Six sculptures — C, D, E, F, G, H — are to be exhibited in rooms 1, 2, and 3 of an art gallery.

- Sculptures C and E may not be exhibited in the same room.
- Sculptures D and G must be exhibited in the same room.
- If sculptures E and F are exhibited in the same room, no other sculpture may be exhibited in that room.
- At least one sculpture must be exhibited in each room, and no more than three sculptures may be exhibited in any room.

If sculpture D is exhibited in room 3 and sculptures E and F are exhibited in room 1, which of the following may be true?

A. Sculpture C is exhibited in room 1.
B. Sculpture H is exhibited in room 1.
C. Sculpture G is exhibited in room 2.
D. Sculptures C and H are exhibited in the same room.
E. Sculptures G and F are exhibited in the same room.

Scope needs to be resolved

At least one sculpture must be exhibited in each room.

The same sculpture in each room?

No more than three sculptures may be exhibited in any room.

Reading 1: For every room, there are no more than three sculptures exhibited in it.

Reading 2: Only three or fewer sculptures are exhibited (the rest are not shown).

Reading 3: Only a certain set of three or less sculptures may be exhibited in any room (for the other sculptures there are restrictions in allowable rooms).

- Some readings will be ruled out by being uninformative or by contradicting other statements

- Otherwise we must be content with distributions over scope-resolved semantic forms

Multiple scope bearers

Most actors tell a joke in every film.

With three scope bearers, there can be 6 possible readings:

MEA	Most actors, in every film, tell a joke (not necessarily the same one each time).
MAE	Most actors have some joke that they tell in every film (the same one each time).
EMA	In every film, most the actors tell a joke (not necessarily the same one).
EAM	In every film, some joke is told by most of the actors (the same one).
AME	There is some joke which most actors tell in each of their films (the same one).
AEM	There is some joke which is told in every film, and by most of the actors.

With *n* scope bearers, up to *n*! possible readings!

It's not just about quantifiers!

- Negation

  All these books are not interesting.
  All that glitters is not gold.

- Modals

  You may not smoke.
  You may not succeed.

- Propositional attitudes

  He claimed he had read each book.

- Implicative verbs

  He failed to answer one question.

Scope generation vs. scope prediction

- Scope generation: enumerating the possible scope readings for a sentence
  - Without (much) regard to plausibility or naturalness

- Scope prediction: determining the relative accessibility of different scope readings
  - By identifying a single reading as the most likely
  - Or by ranking the readings
  - Or by assigning probabilities to each reading

- Most work on scope ambiguity in linguistics has focused on scope generation, not scope prediction!

Classic example

Every man loves a woman

Reading 1: the women may be different
$$\forall x \, (man(x) \rightarrow \exists y \, (woman(y) \wedge love(x,y)))$$

Reading 2: there is one particular woman
$$\exists y \, (woman(y) \wedge \forall x \, (man(x) \rightarrow love(x,y)))$$

What does our system do?
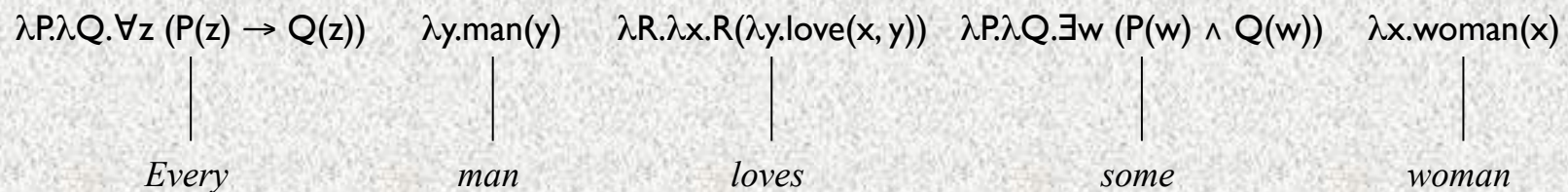(The one we developed last week, with the lambdas.)

Scope ambiguity in action

$(\lambda Q.\forall z\ (man(z) \rightarrow Q(z)))(\lambda x.\exists w\ (woman(w) \wedge love(x, w)))$
$= \forall z\ (man(z) \rightarrow (\lambda x.\exists w\ (woman(w) \wedge love(x, w)))(z))$
$= \forall z\ (man(z) \rightarrow \exists w\ (woman(w) \wedge love(z, w)))$

$(\lambda R.\lambda x.R(\lambda y.love(x, y)))(\lambda Q.\exists w\ (woman(w) \wedge Q(w)))$
$= \lambda x.(\lambda Q.\exists w\ (woman(w) \wedge Q(w)))(\lambda y.love(x, y))$
$= \lambda x.\exists w\ (woman(w) \wedge (\lambda y.love(x, y))(w))$
$= \lambda x.\exists w\ (woman(w) \wedge love(x, w))$

$(\lambda P.\lambda Q.\forall z\ (P(z) \rightarrow Q(z)))(\lambda y.man(y))$
$= \lambda Q.\forall z\ ((\lambda y.man(y))(z) \rightarrow Q(z))$
$= \lambda Q.\forall z\ (man(z) \rightarrow Q(z))$

$(\lambda P.\lambda Q.\exists w\ (P(w) \wedge Q(w)))(\lambda x.woman(x))$
$= \lambda Q.\exists w\ ((\lambda x.woman(x))(w) \wedge Q(w))$
$= \lambda Q.\exists w\ (woman(w) \wedge Q(w))$

$\lambda P.\lambda Q.\forall z\ (P(z) \rightarrow Q(z))$ $\lambda y.man(y)$ $\lambda R.\lambda x.R(\lambda y.love(x, y))$ $\lambda P.\lambda Q.\exists w\ (P(w) \wedge Q(w))$ $\lambda x.woman(x)$

Every   *man*   *loves*   *some*   *woman*

The same thing, using nltk.sem.logic

```
>>> import nltk
>>> from nltk.sem import logic
>>> logic.demo()
>>> parser = logic.LogicParser(type_check=True)

>>> man = parser.parse("\ y.man(y)")
>>> woman = parser.parse("\ x.woman(x)")
>>> love = parser.parse("\ R x.R(\ y.love(x,y))")
>>> every = parser.parse("\ P Q.all x.(P(x) -> Q(x))")
>>> some = parser.parse("\ P Q.exists x.(P(x) & Q(x))")

>>> every(man).simplify()
<LambdaExpression \Q.all x.(man(x) -> Q(x))>

>>> love(some(woman)).simplify()
<LambdaExpression \x.exists z.(woman(z) & love(x, z))>

>>> every(man)(love(some(woman))).simplify()
<AllExpression all x.(man(x) -> exists z.(woman(z) & love(x, z)))>
```

Is this a real problem?

- Our semantic construction fails to generate both readings

- Consider again: *Every man loves some woman.*
  - Note that the wide-scope universal reading is entailed by the wide-scope existential reading
  - In some sense, this means that the stronger reading is superfluous

- Do we really need to handle quantifier scope ambiguities?
  - Yes: straightforward semantic construction doesn't always yield the weakest reading:
    - Consider: *Some man loves every woman.*
  - Yes: quantifier scope interactions sometimes give rise to logically independent readings

Quantifier raising [Montague, 1973]

- Our current semantic construction method allows only one semantic representation per syntactic representation

- So, create more syntactic representations:
  - Add a new grammar rule (Quantifer Raising) that allows us to move NPs in the syntax tree
  - Change semantic construction rules so the desired semantic representations can be computed from the modified trees

Quantifier raising mechanics

- Allow combination with an 'indexed pronoun' instead of a quantifying NP
    - Indexed pronouns are placeholders for a quantifying NP

- Rule of quantification (QR)
    - When the desired scope has been assembled, combine with quantifying NP
    - In the semantic representation, abstract over the variable corresponding to the indexed pronoun

Quantifier raising in action

S: $(\lambda Q.\exists w \ (woman(w) \wedge Q(w)))(\lambda v_1.\forall z \ (man(z) \rightarrow love(z, v_1)))$
$= \exists w \ (woman(w) \wedge (\lambda v_1.\forall z \ (man(z) \rightarrow love(z, v_1)))(w))$
$= \exists w \ (woman(w) \wedge \forall z \ (man(z) \rightarrow love(z, w)))$

S: $(\lambda Q.\forall z \ (man(z) \rightarrow Q(z)))(\lambda x.love(x, v_1))$
$= \forall z \ (man(z) \rightarrow (\lambda x.love(x, v_1))(z))$
$= \forall z \ (man(z) \rightarrow love(z, v_1))$
➡ $\lambda v_1.\forall z \ (man(z) \rightarrow love(z, v_1))$

VP: $(\lambda R.\lambda x.R(\lambda y.love(x, y)))(\lambda P. P(v_1))$
$= \lambda x.(\lambda P. P(v_1))(\lambda y.love(x, y))$
$= \lambda x.(\lambda y.love(x, y))(v_1)$
$= \lambda x.love(x, v_1)$

NP: $\lambda Q.\forall z \ (man(z) \rightarrow Q(z))$　　TV: $\lambda R.\lambda x.R(\lambda y.love(x, y))$　　NP: $\lambda P. P(v_1)$　　NP: $\lambda Q.\exists w \ (woman(w) \wedge Q(w))$

Every man　　　　　　　*loves*　　　　　　[*her*]　　　　　　*some woman*

Quantifier raising: does it work?

- QR allows us to have more than one semantic representation per sentence

- But it does so by treating quantifier scope ambiguity (a *semantic* phenomenon) in terms of unmotivated *syntactic* ambiguity

- It also requires grammar rules with semantic operations other than just function application:
  - Intermediate formulas introduce free (indexed) variables
  - QR involves lambda abstraction over these indexed variables

slide adapted from David Ahn

Cooper storage [Cooper, 1975]

- Storage methods move the key component of Montague's approach from the syntax to semantic construction

- Main ideas:
  - Associate each node of a parse tree with a store
  - A store contains a 'core' semantic representation together with the quantifiers associated with nodes lower in the tree
  - After the sentence is parsed, the store can be used to generate scoped representations
  - The order in which elements are retrieved from the store determines the relative quantifier scopes
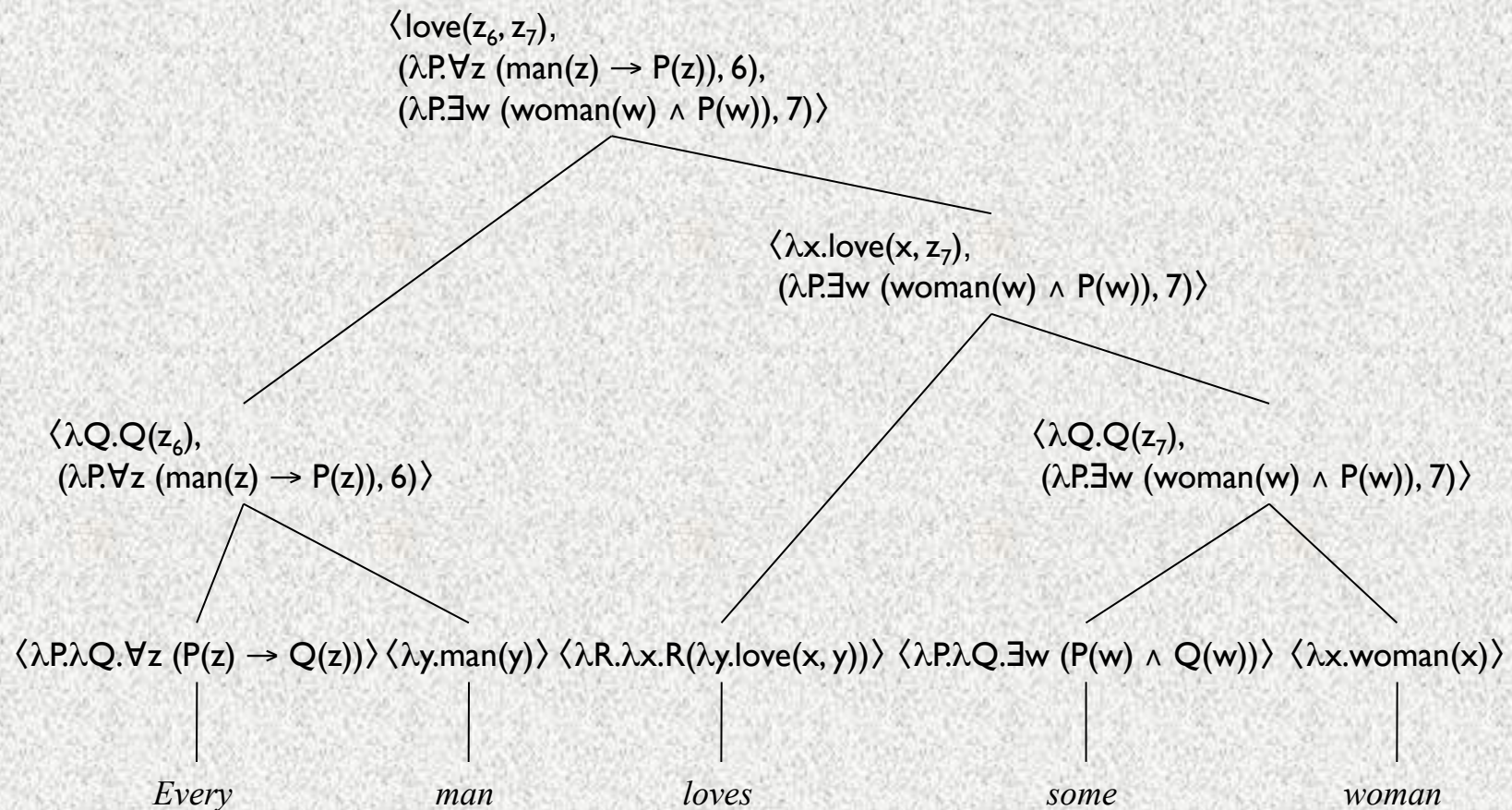
Cooper storage, formally

- A store is an n-place sequence, delimited by angle brackets

- The first item of the sequence is the core semantic representation

- Subsequent items in the sequence are pairs (β, i), where β is the representation of an NP and i is an index

- An index is a label which picks out a free variable in the core semantic representation

Cooper storage: the storage rule

- How do we use stores for semantic construction?

- Quantified NPs are free to 're-package' the information that the store contains:

- If the store $\langle \phi, (\beta, j), \ldots, (\beta', k) \rangle$ is a semantic representation for a quantified NP, then the store $\langle \lambda P.P(z_i), (\phi, i), (\beta, j), \ldots, (\beta', k) \rangle$, where i is a unique index, is also a representation for that NP

Cooper storage in action

⟨love(z_6, z_7),
(λP.∀z (man(z) → P(z)), 6),
(λP.∃w (woman(w) ∧ P(w)), 7)⟩

⟨λx.love(x, z_7),
(λP.∃w (woman(w) ∧ P(w)), 7)⟩

⟨λQ.Q(z_6),
(λP.∀z (man(z) → P(z)), 6)⟩

⟨λQ.Q(z_7),
(λP.∃w (woman(w) ∧ P(w)), 7)⟩

⟨λP.λQ.∀z (P(z) → Q(z))⟩ ⟨λy.man(y)⟩ ⟨λR.λx.R(λy.love(x, y))⟩ ⟨λP.λQ.∃w (P(w) ∧ Q(w))⟩ ⟨λx.woman(x)⟩

Every   *man*   *loves*   *some*   *woman*
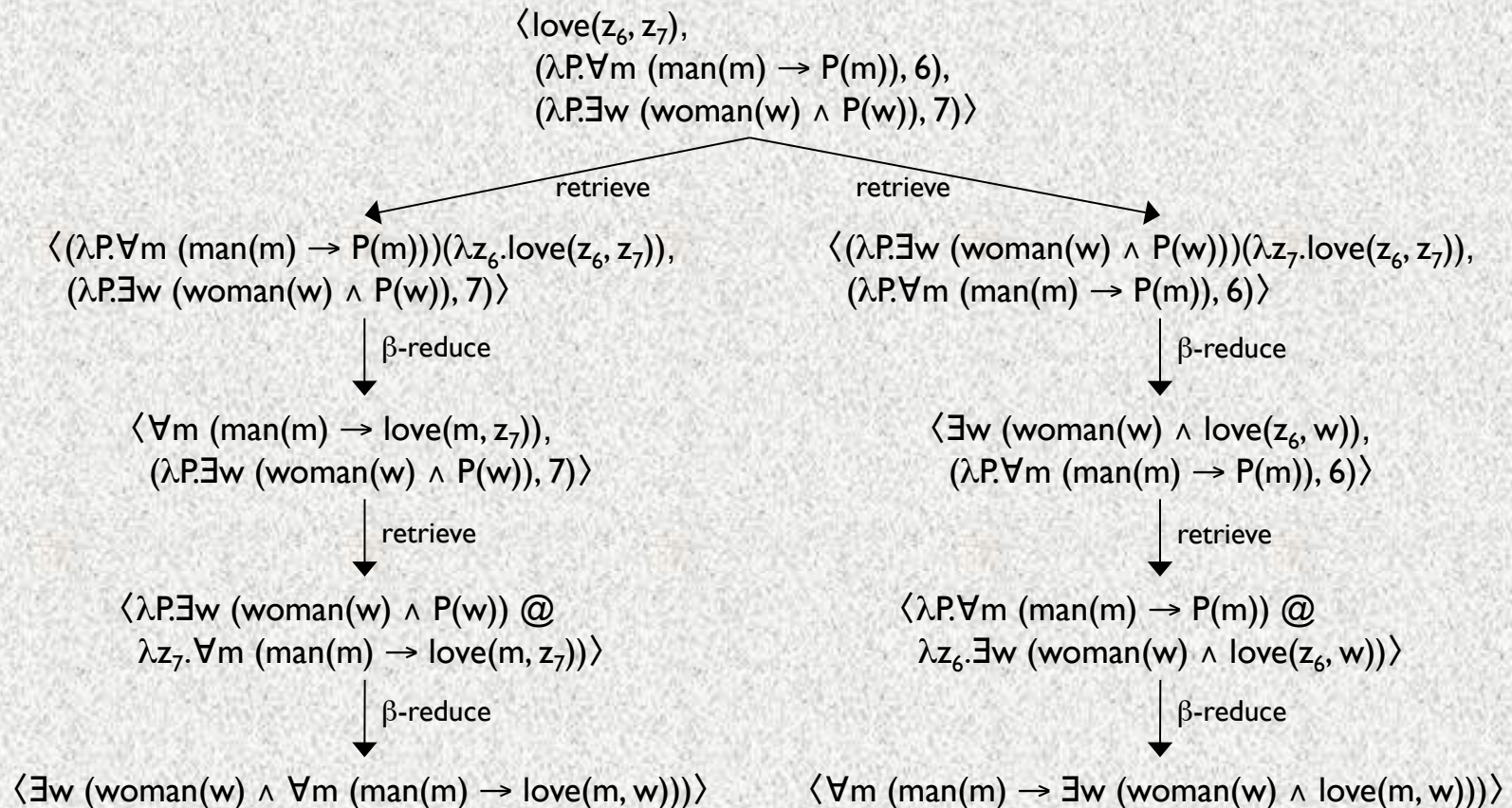
Cooper storage: retrieval

- How do we get ordinary scoped first-order representations out of storage?
  - Remove one of the indexed binding operators from the store
  - Combine it with the core representation
  - Take the result to be the new core representation
  - Continue until there are no more elements in storage (apart from the core representation)

Cooper storage: the retrieval rule

Let σ_1 and σ_2 be (possibly empty) sequences of binding operators. If the store $\langle \phi, \sigma_1, (\beta, i), \sigma_2 \rangle$ is associated with an expression of category S, then the store $\langle \beta(\lambda z_i.\phi), \sigma_1, \sigma_2 \rangle$ is also associated with this expression.

Cooper storage: retrieval example

$\langle love(z_6, z_7),$
$(\lambda P.\forall m\ (man(m) \rightarrow P(m)), 6),$
$(\lambda P.\exists w\ (woman(w) \wedge P(w)), 7)\rangle$

retrieve   retrieve

$\langle (\lambda P.\forall m\ (man(m) \rightarrow P(m)))(\lambda z_6.love(z_6, z_7)),$
$(\lambda P.\exists w\ (woman(w) \wedge P(w)), 7)\rangle$

β-reduce

$\langle \forall m\ (man(m) \rightarrow love(m, z_7)),$
$(\lambda P.\exists w\ (woman(w) \wedge P(w)), 7)\rangle$

retrieve

$\langle \lambda P.\exists w\ (woman(w) \wedge P(w))\ @$
$\lambda z_7.\forall m\ (man(m) \rightarrow love(m, z_7))\rangle$

β-reduce

$\langle \exists w\ (woman(w) \wedge \forall m\ (man(m) \rightarrow love(m, w))))\rangle$

$\langle (\lambda P.\exists w\ (woman(w) \wedge P(w)))(\lambda z_7.love(z_6, z_7)),$
$(\lambda P.\forall m\ (man(m) \rightarrow P(m)), 6)\rangle$

β-reduce

$\langle \exists w\ (woman(w) \wedge love(z_6, w)),$
$(\lambda P.\forall m\ (man(m) \rightarrow P(m)), 6)\rangle$

retrieve

$\langle \lambda P.\forall m\ (man(m) \rightarrow P(m))\ @$
$\lambda z_6.\exists w\ (woman(w) \wedge love(z_6, w))\rangle$

β-reduce

$\langle \forall m\ (man(m) \rightarrow \exists w\ (woman(w) \wedge love(m, w))))\rangle$

Cooper storage: does it work?

- We are allowed to retrieve quantifiers in any order

- The only safety net is the use of co-indexed variables

- Is this really safe?

  - Consider: *Mia knows every owner of a hash bar.*

  - In wide-scope universal reading, there is a free variable:
    $\forall y \, (\text{own}(y, z_1) \rightarrow \exists x \, (\text{hashbar}(x) \wedge \text{know}(\text{mia}, y)))$

  - Solution: nested stores [Keller, 1988]

Combinatorial explosion

Most actors tell a joke in every film.

As we saw earlier, there are 6 possible readings here.

Having *n* scope bearers means up to *n*! possible readings!

Most politicians can fool most voters on most issues
most of the time, but no politician can fool every voter
on every single issue all of the time.

Hobbs estimates that there are 600 possible readings here.

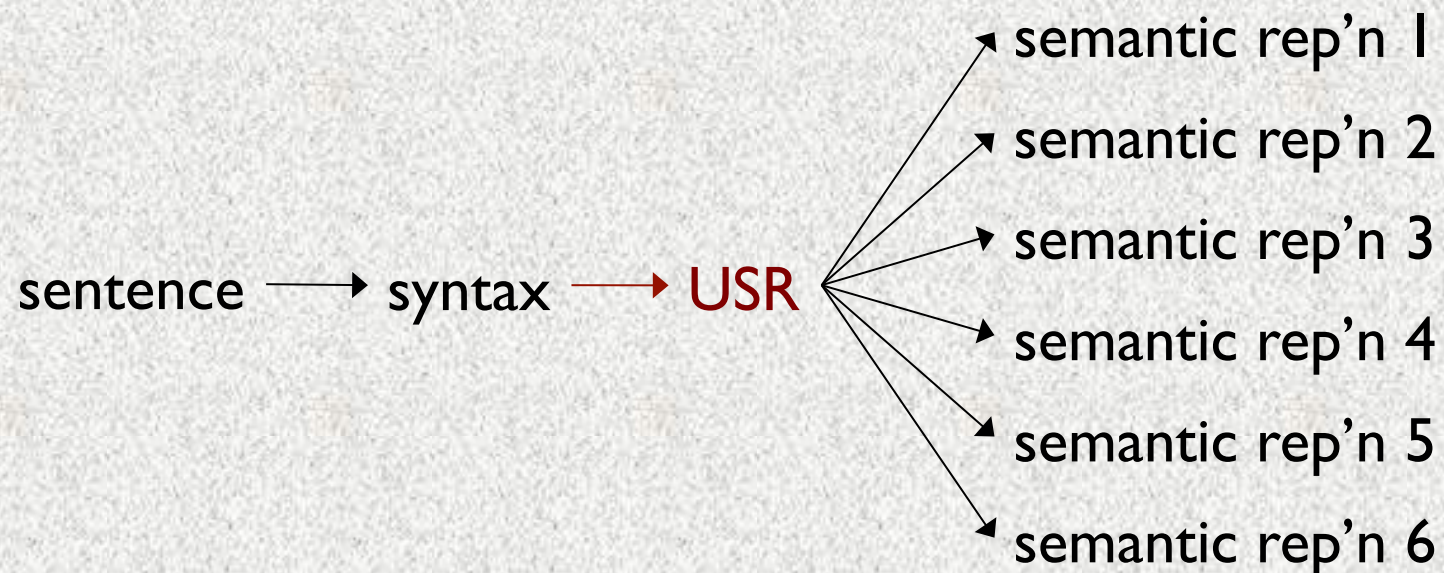Please enumerate them.

Explosion in semantic construction

sentence → syntax

- semantic rep'n 1
- semantic rep'n 2
- semantic rep'n 3
- semantic rep'n 4
- semantic rep'n 5
- semantic rep'n 6
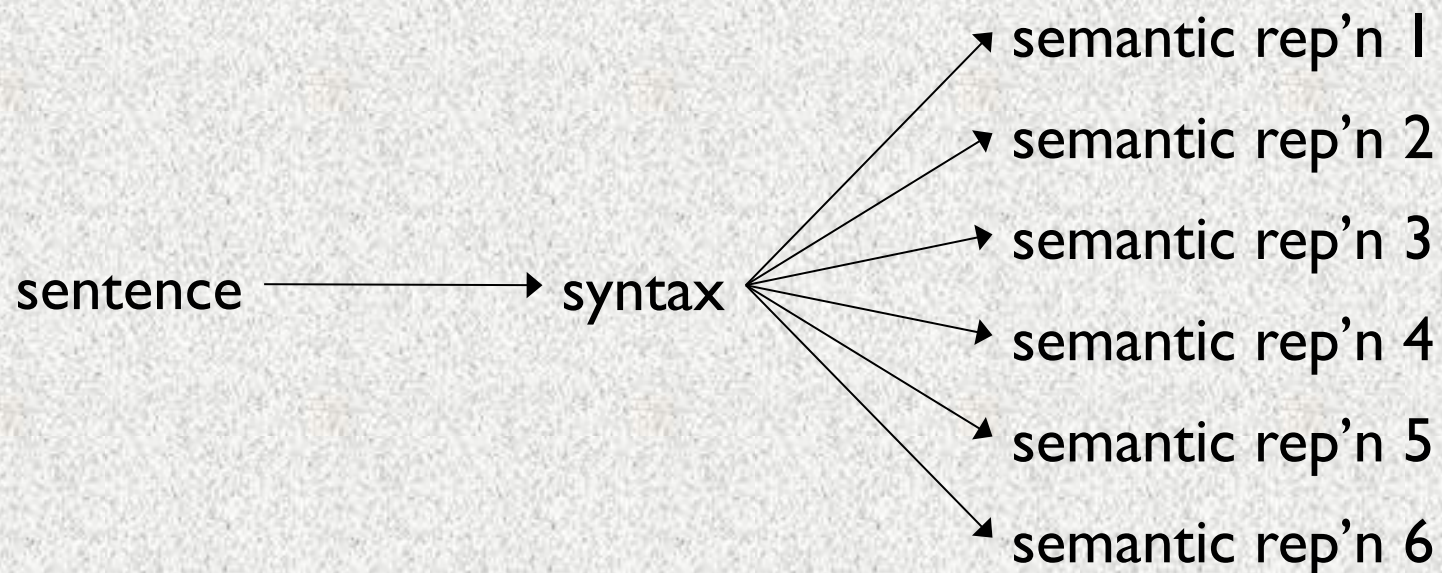
Enumerating readings is expensive

- We'd like to avoid enumerating these many readings.

- Most of the readings were not meant by the speaker.

- Do people enumerate readings of a scope ambiguity?

Scope underspecification

- Given the exponential number of possible readings, we'd like to postpone enumeration for as long as we can.

- Take a single syntactic analysis and derive a single underspecified semantic representation from it.

- Possibly perform inferences on underspecified descriptions to remove unwanted readings.

- Then enumerate readings from description as needed.

Underspecified semantic representations

sentence $\longrightarrow$ syntax $\longrightarrow$ USR

- semantic rep'n 1
- semantic rep'n 2
- semantic rep'n 3
- semantic rep'n 4
- semantic rep'n 5
- semantic rep'n 6

Underspecification

- Storage methods (e.g., Cooper storage) provide a limited sort of underspecification:
    - Represent all possible meanings compactly
    - But don't allow additional constraints on possible readings to be expressed

- Modern approaches to underspecification take a more abstract view of semantic representation:
    - Treat semantic representations as objects
    - Use a meta-level representation to encode information about semantic representations

Constraints on possible scopings

Some scopings are prohibited by syntactic constraints.

Anyone *who skips* *at most two* *sessions will get* *a* *prize.*

There are three scope bearers here.

How many possible readings?

Which reading(s) are not possible?

Underspecification formalisms

QLF
(Alshawi & Crouch 92)

Hole semantics
(Bos 96)

Dominance constraints/graphs
(Koller et al. 00)

UDRT
(Reyle 93)

MRS
(Copestake et al. 99)

LDG
(Muskens 95)

CLLS
(Egg et al. 98)

Dominance graphs

- Semantic representations are formulas of first-order logic.

- Read these formulas as trees.

- Describe these trees using graphs into which they can be embedded.

- Use special graph edges to represent variable binding (not shown here).

- Equivalent to normal dominance constraints, i.e. both graph view and logic view available.

Describe trees using graphs

"Every student reads a book."

$\exists y\ book(y) \land \forall x.student(x) \rightarrow read(x,y)$

$\forall x.student(x) \rightarrow \exists y\ book(y) \land read(x,y)$
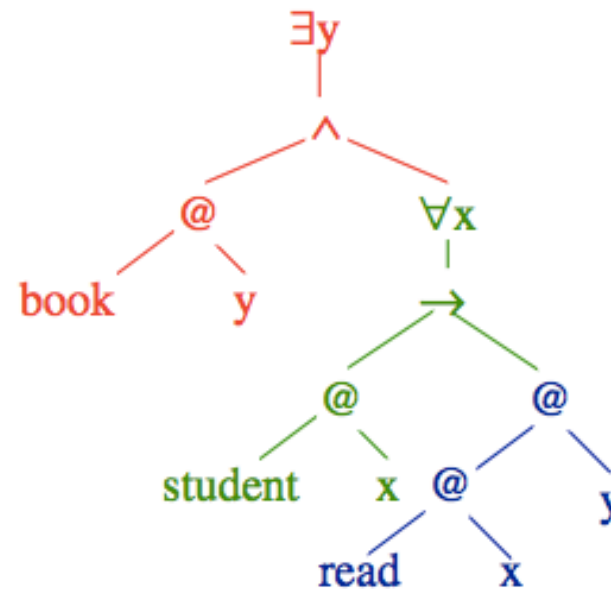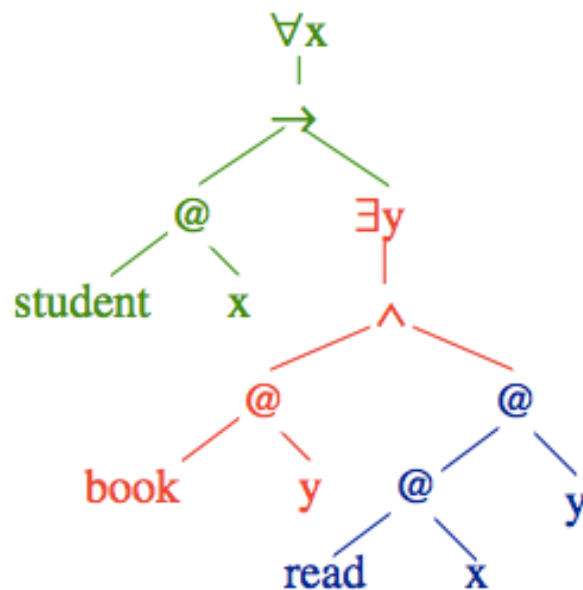
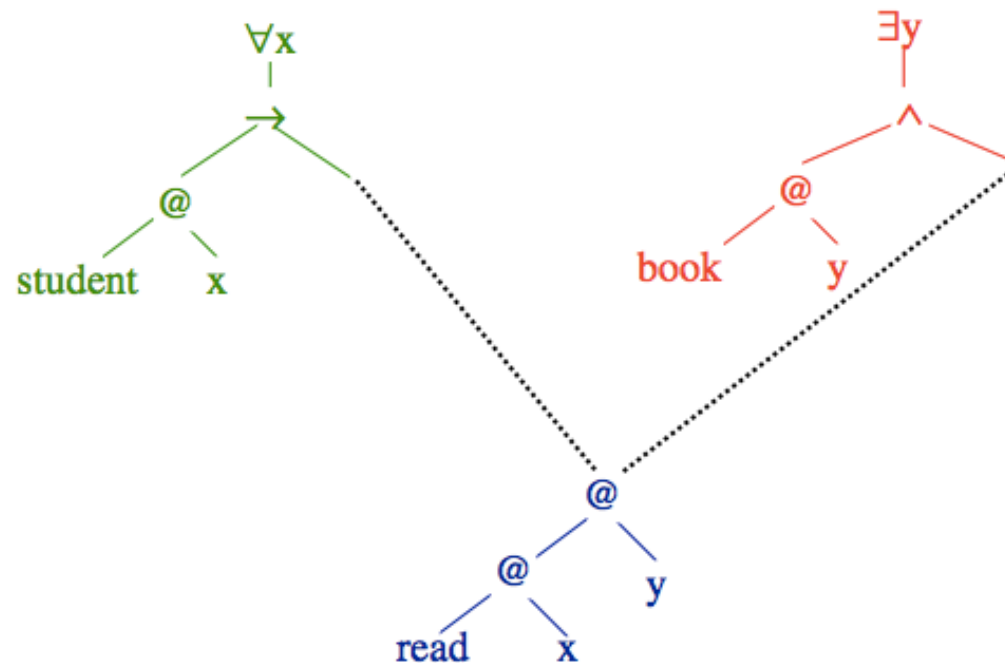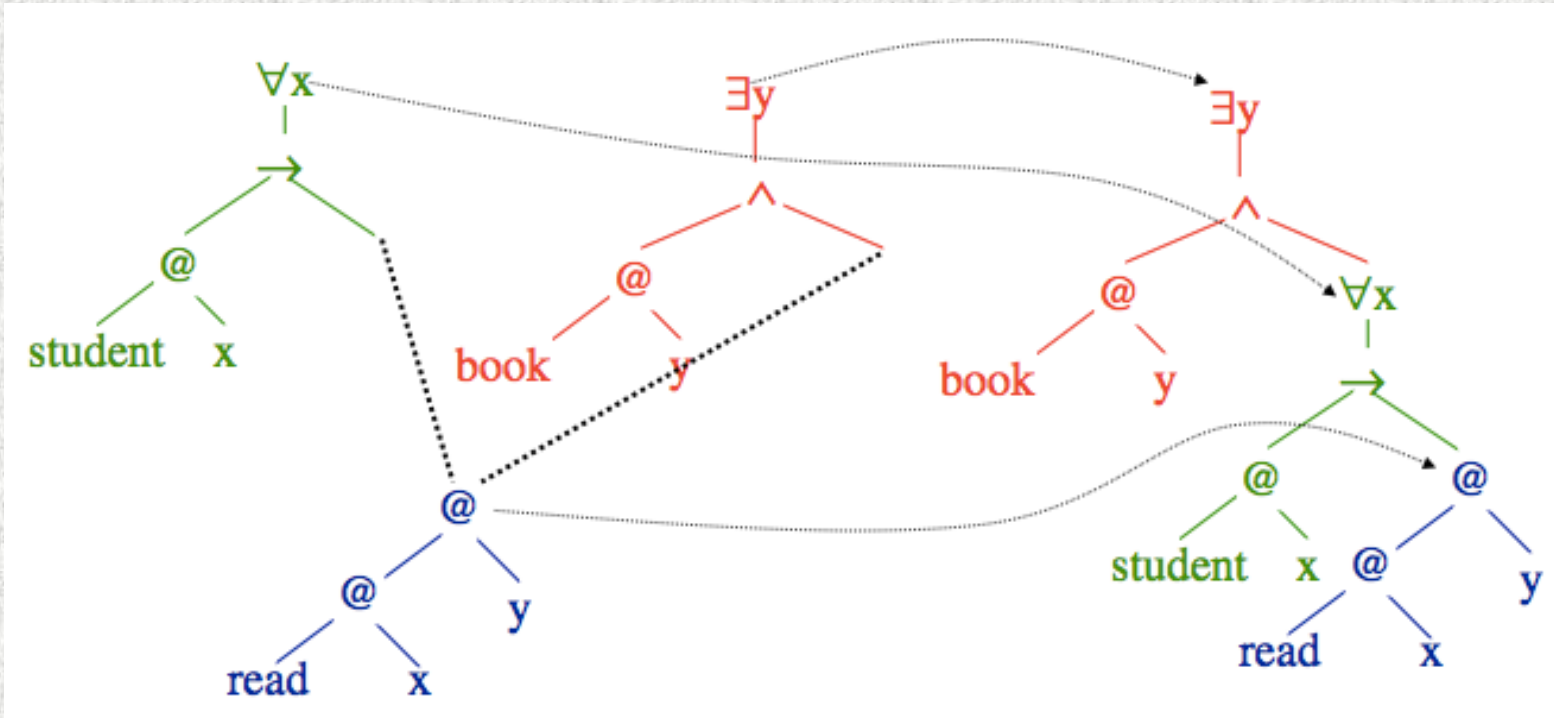# Express constraints with special edges

"Every student reads a book."
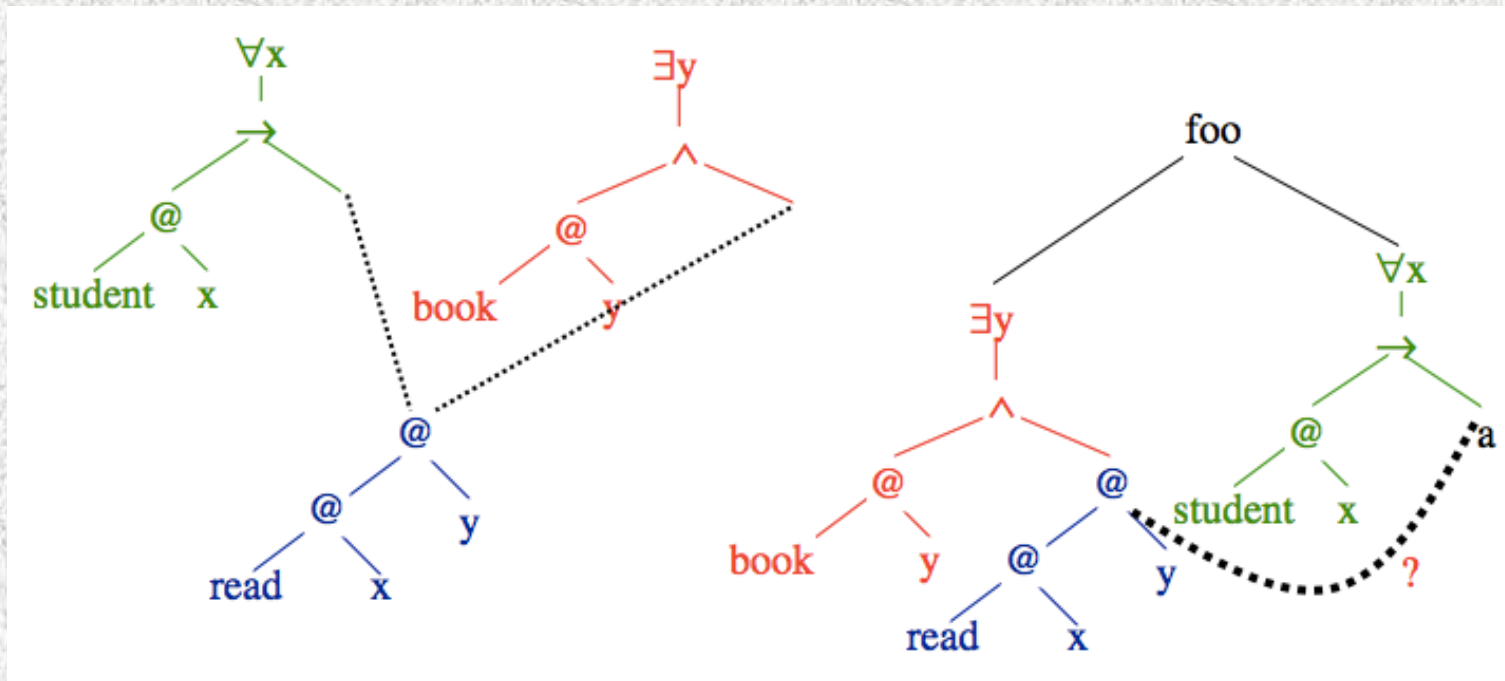$\exists y\ book(y) \wedge \forall x.student(x) \rightarrow read(x,y)$
$\forall x.student(x) \rightarrow \exists y\ book(y) \wedge read(x,y)$

A solution to a dominance graph

A different solution

Not a solution

What about scope prediction?

- Let's turn from scope generation to scope prediction.

- What are some heuristics that are often good predictors?
  - subject quantifiers tend to take wide scope
  - universals tend to take wide scope
  - topicalized quantifiers tend to take wide scope
  - "each" tends to take wide scope
  - "most" tends to take wide scope

- Can we put these into a statistical model?

Discussion: Higgins & Sadock 03

- What was their goal?

- What was their method?
  - Data: selection? annotation?
  - Classifiers
  - Features

- What were their results?
  - Good? Convincing? Useful?

What's missing?

OK, this all seems super duper, but … what's missing?

Can we solve these NLU challenges yet?

Why not?

Six sculptures — C, D, E, F, G, H — are to be exhibited in rooms 1, 2, and 3 of an art gallery.

- Sculptures C and E may not be exhibited in the same room.
- Sculptures D and G must be exhibited in the same room.
- If sculptures E and F are exhibited in the same room, no other sculpture may be exhibited in that room.
- At least one sculpture must be exhibited in each room, and no more than three sculptures may be exhibited in any room.

If sculpture D is exhibited in room 3 and sculptures E and F are exhibited in room 1, which of the following may be true?

A. Sculpture C is exhibited in room 1.
B. Sculpture H is exhibited in room 1.
C. Sculpture G is exhibited in room 2.
D. Sculptures C and H are exhibited in the same room.
E. Sculptures G and F are exhibited in the same room.

Yes, hi, I need to book a flight for myself and my husband from Boston to SFO, or Oakland would be OK too. We need to fly out on Friday the 12th, and then I could fly back on Sunday evening or Monday morning, but he won't return until Wednesday the 18th, because he's staying for business. No flights with more than one stop, and we don't want to fly on United because we hate their guts.

Lots of remaining challenges!

- Additional semantic issues, e.g. a general account of coordination, anaphora resolution, semantics of anaphoric adjectives, …

- Treatment of discourse pragmatics, indexicals and the role of context, etc.

- Relies on perfect syntactic analysis! No robustness to bad parsing — or bad / informal / internet English

- Doesn't resolve semantic ambiguities (or, resolves them deterministically, heedless of context)

- No notion of probability / likelihood of various readings

- Hand-built semantic grammars — laborious, error-prone, domain-specific

- No application of machine learning, either for lexicon or for composition rules