Compositional Semantics: Quantification and Underspecification

COMP-550

Oct 31, 2017

Midterm

Thursday, Nov 9, in class, 80 min.

Closed book

Question types:

- M/C
- Short answer
- Problem solving

Expectations

You'll be expected to know:

- Definitions (including formulas)
- How to run algorithms by hand
- Simple derivations
- Simple probability and calculus (up to, say, chain rule)
- Basic linguistic terminology and theory
- Lambda calculus

You won't be expected to remember*:

- Lagrange multipliers
- What the derivative of, say, e^x is.
- Specific lexical rules for translating items into FOL

*But could still be asked to apply or use, given relevant facts

How To Study

Review and redo the exercises we did in class

Learn the definitions of all the technical terms (most of them are in bold or are the titles of slides)

Review your assignments

Do practice problems

Do the practice midterm

List of practice problems from the textbook are on the course website. Note that they are not meant to be comprehensive, but are supplementary.

Outline

Syntax-driven semantic composition

Quantifiers

Generalized quantifiers

Underspecification

Power of Lambda Calculus

They allow us to store partial computations of the MR, as we are composing the meaning of the sentence constituent by constituent.

Whiskers disdained catnip.

disdained	$\lambda x.\lambda y.disdained(y,x)$
disdained catnip	$\big(\lambda x.\lambda y.disdained(y,x)\big)\ catnip$
	$= \lambda y.disdained(y,catnip)$

Whiskers disdained catnip

$$\big(\lambda y.disdained(y,catnip)\big)Whiskers$$

$$= disdained(Whiskers,catnip)$$

Exercises

What is the result of simplifying the following expressions in lambda calculus through beta reduction?

$(\lambda z. z)(\lambda y. y\ y)(\lambda x. x\ a)$

$(((\lambda x. \lambda y. (x\ y))(\lambda y. y))\ w)$

$(\lambda x. x\ x)\ (\lambda y. y\ x)\ z$

Syntax-Driven Semantic Composition

Augment CFG trees with lambda expressions

- Syntactic composition = function application

Semantic attachments:

$A \rightarrow \alpha_1 \ \ldots \alpha_n$ $\{f(\alpha_j.sem, \ldots, \alpha_k.sem)\}$

syntactic composition semantic attachment

Proper Nouns

Proper nouns are FOL constants

$PN \rightarrow COMP550 \qquad \{COMP550\}$

Actually, we will **type-raise** proper nouns

$PN \rightarrow COMP550 \qquad \{\lambda x.x(COMP550)\}$

- It is now a function rather than an argument.
- We will see why we do this.

NP rule:

$NP \rightarrow PN \qquad \{PN.sem\}$

Common Nouns

Common nouns are predicates inside a lambda expression of type $\langle e, t \rangle$

- Takes an entity, tells you whether the entity is a member of that class

$$N \rightarrow student \qquad \{\lambda x. Student(x)\}$$

Let's talk more about common nouns next class when we also talk about quantifiers.

Intransitive Verbs

We introduce an *event variable* e, and assert that there exists a certain event associated with this verb, with arguments.

$$V \rightarrow rules \qquad \{\lambda x. \exists e. Rules(e) \wedge Ruler(e, x)\}$$

Then, composition is

$$S \rightarrow NP\ VP \qquad \{NP.sem(VP.sem)\}$$

Let's derive the representation of the sentence "*COMP-550 rules*"

Neo-Davidsonian Event Semantics

Notice that we have changed how we represent events

Method 1: multi-place predicate

$Rules(x)$

Method 2: Neo-Davidsonian version with event variable

$\exists e. Rules(e) \land Ruler(e, x)$

Reifying the event variable makes things more flexible

- Optional elements such as location and time, passives
- Add information to the event variable about tense, modality

Transitive Verbs

Transitive verbs

$V \rightarrow enjoys$

$\{\lambda w.\lambda z.w(\lambda x.\exists e.Enjoys(e) \wedge Enjoyer(e,z) \wedge Enjoyee(e,x))\}$

$VP \rightarrow V\ NP \qquad \{V.sem(NP.sem)\}$

$S \rightarrow NP\ VP \qquad \{NP.sem(VP.sem)\}$

Exercise: verify that this works with the sentence "*Jackie enjoys COMP-550*"

Quantifiers

Universal quantifiers

- *all, every*

All students like COMP-550.

$\forall x. Student(x) \rightarrow Like(x, \mathrm{COMP{-}550})$

Existential quantifiers

- *a, an, some*

Some/A student likes COMP-550.

$\exists x. Student(x) \wedge Like(x, \mathrm{COMP{-}550})$

Why $\rightarrow$ for the universal quantifier, but $\wedge$ for the existential one?

Russell (1905)'s Definite Descriptions

How to express "*the student*" in FOL?

e.g., *The student took COMP-599.*

Need to enforce three properties:

1. There is an entity who is the student.
2. There is at most one thing being referred to who is a student.
3. The student participates in some predicate, here, "*took COMP-550*".

The King of France is Bald

Property 1 is important. Consider "*The King of France is bald.*"

Solution 1:

- Define a new constant for KING-OF-FRANCE, much like for proper nouns.

FOL MR becomes $Bald(\mathrm{KING{-}OF{-}FRANCE})$

What is the problem with this solution?

Definite Articles

The student took COMP-550:

1. There is an entity who is the student.
2. There is at most one thing being referred to who is a student.
3. The student participates in some predicate.

What is the range of this existential quantifier?

$$\exists x. Student(x) \land \forall y. (Student(y) \rightarrow y = x) \land took(x, \text{COMP}-550)$$

For simplicity, for now, assume took is a predicate, rather than use event variables.

Incorporating into Syntax

Now, let's incorporate this to see how lambda calculus can deal with this compositionally.

Semantic attachment for lexical rule for *every*:

$Det \rightarrow every \qquad \{\lambda P.\lambda Q.\forall x.P(x) \rightarrow Q(x)\}$

What do P and Q represent?

Every Student Likes COMP-550

$Det \rightarrow every \qquad \{\lambda P. \lambda Q. \forall x. P(x) \rightarrow Q(x)\}$

$NP \rightarrow Det\ N \qquad \{Det.sem(N.sem)\}$

Let's do the derivation of *Every student likes COMP-599*.

Recall:

$VP \rightarrow V\ NP \qquad \{V.sem(NP.sem)\}$

$S \rightarrow NP\ VP \qquad \{NP.sem(VP.sem)\}$

$V \rightarrow likes$

$$\{\lambda w. \lambda z. w(\lambda x. \exists e. Likes(e) \wedge Liker(e, z) \wedge Likee(e, x))\}$$

Using explicit event variables again.

Questions and Exercise

What are the lexical rules with semantic attachments for *a*? For *the*?

Come up with the derivation of *COMP-550 likes every student.*

Adjectives

Can we figure out the pattern for adjectives?

student $\lambda x. Student(x)$

smart student $\lambda x. Smart(x) \land Student(x)$

smart ?

Also need an augmented rule for N -> A N

Scopal Ambiguity: Multiple Quantifiers

What are the possible readings for the following?

Every student took a course.

This is known as **scopal ambiguity**.

Scopal Ambiguity

Every student took a course.

every > a

$\forall x. Student(x)$
$\rightarrow (\exists y. Course(y) \wedge \exists e. took(e) \wedge taker(e, x) \wedge takee(e, y))$

a > every

$\exists y. Course(y)$
$\wedge \left(\forall x. Student(x) \rightarrow \exists e. took(e) \wedge taker(e, x) \wedge takee(e, y)\right)$

Would like a way to derive **both** of these readings from the syntax. What would we get with our current method?

Underspecification

Solution: Derive a representation that allows for both readings

Underspecified representation – A meaning representation that can embody all *possible* readings without explicitly enumerating all of them.

Other cases where this is useful:

- We are genuinely missing some information (e.g., the tense information), so we choose not to include it in the meaning representation.

Cooper Storage (1983)

Associate a **store** with each FOL expression that allows both readings to be recovered.

Every student took a course.

$$\exists e.\, took(e) \wedge taker(e, s_1) \wedge takee(e, s_2)$$

$$(\lambda Q.\, \forall x.\, Student(x) \rightarrow Q(x), 1),$$

$$(\lambda Q.\, \exists y.\, Course(y) \wedge Q(y), 2)$$

Recovering the Reading

Once we know which reading we want (e.g., by looking at the context), recover the store:

1. Select order to incorporate quantifiers
2. For each quantifier:
    - Introduce lambda abstraction over the appropriate index variable
    - Do beta-reduction

Example: 1, then 2

Every student took a course.

$$\exists e.\, took(e) \wedge taker(e, s_1) \wedge takee(e, s_2)$$

$$(\lambda Q.\, \forall x.\, Student(x) \rightarrow Q(x), 1),$$

$$(\lambda Q.\, \exists y.\, Course(y) \wedge Q(y), 2)$$

1 first:

$$\big(\lambda Q.\, \forall x.\, Student(x) \rightarrow Q(x)\big)$$

$$\big(\lambda s_1.\, \exists e.\, took(e) \wedge taker(e, s_1) \wedge takee(e, s_2)\big)$$

$= \forall x.\, Student(x) \rightarrow \exists e.\, took(e) \wedge taker(e, x) \wedge takee(e, s_2)$

Then 2:

$$\big(\lambda Q.\, \exists y.\, Course(y) \wedge Q(y)\big)$$

$$\big(\lambda s_2.\, \forall x.\, Student(x) \rightarrow \exists e.\, took(e) \wedge taker(e, x) \wedge takee(e, s_2)\big)$$

$= \exists y.\, Course(y) \wedge \forall x.\, Student(x) \rightarrow \exists e.\, took(e) \wedge taker(e, x) \wedge takee(e, y)$

Compositional Rules

We also need new rules with semantic attachments for our quantifiers:

- Composing quantifier with N is now modifying the *inside* part of a store
- An NP is now introduces a new index variable, which is wrapped in a lambda expression

A Course

NP $\lambda u.u(s_1), (\lambda Q.\exists x.Course(x) \wedge Q(x), 1)$

Det *a* $(\lambda P.\lambda Q.\exists x.P(x) \wedge Q(x))$

N *course* $(\lambda x.Course(x))$

What is the semantic attachment for NP -> Det N? Use .sem.store to access the store.

Exercise

Finish the derivation for the underspecified representation of *Every student took a course.*

Recall:

$VP \rightarrow V\ NP \qquad \{V.sem(NP.sem)\}$

$S \rightarrow NP\ VP \qquad \{NP.sem(VP.sem)\}$

$V \rightarrow took$

$\{\lambda w.\lambda z.w(\lambda x.\exists e.Took(e) \wedge Taker(e,z) \wedge Takee(e,x))\}$

$Det \rightarrow every \qquad \{(\lambda P.\lambda Q.\forall x.P(x) \rightarrow Q(x))\}$

$N \rightarrow student \qquad \{\lambda x.Student(x)\}$

Question

How would we disambiguate between the possible readings?