Compositional Semantics

CMSC 723 / LING 723 / INST 725

MARINE CARPUAT
marine@cs.umd.edu
Last week... Intro to Semantics

– Meaning representations
  • motivated by semantic processing
  • for specific applications

– 2 approaches to semantic processing
  • complete FOL representation
  • vs. shallow information extraction
Semantic Analysis: 2 approaches

• Compositional Analysis
  – Complete analysis
  – Create a First Order Logic representation that accounts for all the entities, roles and relations present in a sentence

• Information Extraction
  – Superficial analysis
  – Pulls out only the entities, relations and roles that are of interest to the consuming application.
Today... Compositional Semantics

• Representing the meaning of declarative sentences using FOL

• From syntax to semantics
FIRST ORDER LOGIC
First Order Logic as Representational Framework

Allows for...

– The analysis of truth conditions
  • Allows us to answer yes/no questions

– Supports the use of variables
  • Allows us to answer questions through the use of variable binding

– Supports inference
  • Allows us to answer questions that go beyond what we know explicitly
FOL sufficient for many natural language inferences

- All blips are foos.
- Blop is a blip.
- Blop is a foo.

- Mozart was born in Salzburg.
- Mozart was born in Vienna.
- No, that can’t be. These are different cities.
First Order Logic

• **Syntax**: what is the language of well-formed formulas?

• **Semantics**: what is the interpretation of a well-formed formula?

• **Inference rules and algorithms**: how can we reason with predicate logic? (not covered in this class)
A Model of “World of Nearby Restaurants” using First Order Logic

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Formula</strong></td>
<td>$\text{AtomicFormula}$</td>
</tr>
<tr>
<td></td>
<td>$\text{Formula Connective Formula}$</td>
</tr>
<tr>
<td></td>
<td>$\text{Quantifier Variable, }\ldots\text{ Formula}$</td>
</tr>
<tr>
<td></td>
<td>$\neg \text{Formula}$</td>
</tr>
<tr>
<td></td>
<td>$(\text{Formula})$</td>
</tr>
<tr>
<td><strong>AtomicFormula</strong></td>
<td>$\text{Predicate(Term, }\ldots\text{)}$</td>
</tr>
<tr>
<td><strong>Term</strong></td>
<td>$\text{Function(Term, }\ldots\text{)}$</td>
</tr>
<tr>
<td></td>
<td>$\text{Constant}$</td>
</tr>
<tr>
<td></td>
<td>$\text{Variable}$</td>
</tr>
<tr>
<td><strong>Connective</strong></td>
<td>$\land</td>
</tr>
<tr>
<td><strong>Quantifier</strong></td>
<td>$\forall</td>
</tr>
<tr>
<td><strong>Constant</strong></td>
<td>$A</td>
</tr>
<tr>
<td><strong>Variable</strong></td>
<td>$x</td>
</tr>
<tr>
<td><strong>Predicate</strong></td>
<td>$\text{Serves}</td>
</tr>
<tr>
<td><strong>Function</strong></td>
<td>$\text{LocationOf}</td>
</tr>
</tbody>
</table>

See Textbook Section 17.3 for details
Predicate Logic Expressions

- **Terms**: refer to entities, objects in the worlds
- **Predicates**: refer to relations or properties
- **Formulas**: can be true or false
Formulas

- **Atomic**: predicate applied to terms
  
- **Complex**: constructed recursively by negation, connective, quantifiers

- **Interpretation**: either true or false
FOL Models

- A model consists of
  - a domain i.e., a set of entities
  - interpretation of terms
  - Unary predicates that define (sub)sets of entities
  - N-ary predicates that define sets of n-ary tuples of entities
Not all of natural language can be expressed in FOL

• Tense
  – It was hot yesterday.
  – I will go to DC tomorrow.

• Modals
  – You can go to DC from here.

• Other kinds of quantifiers
  – Most students hate 8am lectures.
More examples... How would you represent them in FOL?

• Alice is a student

• All students take at least one class

• There is a class that all students take
COMPOSITIONAL SEMANTICS
Compositional Semantic Analysis

• Semantic analysis
  – the process of taking in some linguistic input and assigning a meaning representation to it.
  
  – Lot of different ways to do this that make more or less (or no) use of syntax

  – We’ll discuss one approach that assumes that syntax does matter
    • The compositional rule-to-rule approach
Principle of Compositionality

• The meaning of a whole is derived from the meanings of the parts

• What parts?
  – The constituents of the syntactic parse of the input

• What could it mean for a part to have a meaning?
Compositional Analysis: use syntax to guide semantic analysis

\[ S \exists e \text{Liking}(e) \land \text{Liker}(e, \text{Franco}) \land \text{Liked}(e, \text{Frasca}) \]

```
NP          Verb          NP
Franco      likes        Frasca
```

```
NP          VP
```

Augmented Rules

• We’ll accomplish this by attaching semantic formation rules to our syntactic CFG rules

• Abstractly

\[ A \rightarrow \alpha_1 \ldots \alpha_n \quad \{ f (\alpha_1.\text{sem}, \ldots \alpha_n.\text{sem}) \} \]

– This should be read as: “the semantics we attach to A can be computed from some function applied to the semantics of A’s parts.”
Example

- Easy parts...
  - NP -> PropNoun
  - PropNoun -> Frasca
  - PropNoun -> Franco

- Attachments
  - {PropNoun.sem}
  - {Frasca}
  - {Franco}
Example

- S → NP VP
- VP → Verb NP
- Verb → likes

\[ \lambda x \lambda y \exists e \text{Liking}(e) \land \text{Likker}(e, y) \land \text{Liked}(e, x) \]
Lambda Forms & Lambda Reductions

• A simple addition to FOL

  – Take a FOL formula with variables in it that are to be bound.

  \[ \lambda x. P(x) \]

  – Allow those variables to be bound by treating the lambda form as a function with formal arguments.

  \[ \lambda x. P(x)(\text{Franco}) \]

  \[ P(\text{Franco}) \]
Compositional semantics by lambda application
Lambda Reductions

\[ \lambda x \lambda y \exists e \mathit{Liking}(e) \land \mathit{Likер}(e, y) \land \mathit{Liked}(e, x)(\text{Frasca}) \]

\[ \lambda y \exists e \mathit{Liking}(e) \land \mathit{Likер}(e, y) \land \mathit{Liked}(e, \text{Frasca}) \]

\[ \lambda y \exists e \mathit{Liking}(e) \land \mathit{Likер}(e, y) \land \mathit{Liked}(e, \text{Frasca})(\text{Franco}) \]

\[ \exists e \mathit{Liking}(e) \land \mathit{Likер}(e, \text{Franco}) \land \mathit{Liked}(e, \text{Frasca}) \]
Complications

• Of course, that’s the simplest possible example.

• Making it work for harder cases is more involved...
  – Mismatches between the syntax and semantics
Complications: Complex NPs

– The previous examples simplified things by only dealing with constants (FOL Terms).

– What about...
  • A menu
  • Every restaurant
  • Not every waiter
  • Most restaurants
Quantifiers

• Last winter, during the storm...

  – Every restaurant closed.

\[\forall x \text{ Restaurant}(x) \Rightarrow \exists e \text{ Closed}(e) \land \text{ClosedThing}(e, x)\]
Quantified NPs

• So from a compositional point of view, what should the semantic fragment for “every restaurant” look like

\[ \forall x \text{ Restaurant}(x) \]

– Hint: this isn’t it yet...
Quantifiers

• Roughly “every” in an NP like this is used to stipulate something about every member of the class:
  – The NP is specifying the class.
  – the VP is specifying the thing stipulate....

So the NP can be viewed as the following template:

\[ \forall x \text{Restaurant}(x) \Rightarrow Q(x) \]
Quantifiers

• But that’s not combinable with anything so wrap a lambda around it...

$$\lambda Q. \forall x \text{Restaurant}(x) \Rightarrow Q(x)$$

• This requires a change to the kind of things that we’ll allow lambda variables to range over...
  – Now its both FOL predicates and terms.
Rules

\[
NP \rightarrow \text{Det Nominal} \quad \{\text{Det.Sem(Nominal.Sem)}\}
\]

\[
\text{Det} \rightarrow \text{every} \quad \{\lambda P.\lambda Q. \forall x \; P(x) \Rightarrow Q(x)\}
\]

\[
\text{Nominal} \rightarrow \text{Noun} \quad \{\text{Noun.sem}\}
\]

\[
\text{Noun} \rightarrow \text{restaurant} \quad \{\lambda x.\text{Restaurant}(x)\}
\]
Example

\[ \lambda P. \lambda Q. \forall x \ P(x) \Rightarrow Q(x)(\lambda x. \text{Restaurant}(x)) \]

\[ \lambda Q. \forall x \ \lambda x. \text{Restaurant}(x)(x) \Rightarrow Q(x) \]

\[ \lambda Q. \forall x \ \text{Restaurant}(x) \Rightarrow Q(x) \]
Every Restaurant Closed
Grammar Engineering

• Remember:
  – in the rule-to-rule approach we’re designing separate semantic attachments for each grammar rule
• So we now have to check to see if things still work with the rest of the grammar!

• Two places to revise...
  – The S rule
    • S --> NP VP VP.Sem(NP.Sem)
  – Simple NP’s like proper nouns...
    • Proper-Noun --> Sally Sally
S Rule

• We were applying the semantics of the VP to the semantics of the NP... Now we’re swapping that around
  – S --> NP VP  NP.Sem(VP.Sem)
Every Restaurant Closed

\[ \forall x \text{Restaurant}(x) \Rightarrow \exists e \text{Closed}(e) \land \text{ClosedThing}(e,x) \]

\[ \lambda Q. \forall x \text{Restaurant}(x) \Rightarrow Q(x) \]

\[ \lambda x. \exists e \text{Closed}(e) \land \text{ClosedThing}(e,x) \]

\[ \lambda P. \lambda Q. \forall x P(x) \Rightarrow Q(x) \]

\[ \lambda x. \text{Restaurant}(x) \]

\text{Every restaurant is closed.}
Simple NP fix

• Now semantics of proper nouns need to be a little more complex.
  – E.g., $\lambda x \text{Franco}(x)$
Revised

• Now all these examples should work
  – *Every restaurant closed.*
  – *Sunflower closed.*

• What about?
  – *A restaurant closed.*

• This rule stays the same
  – NP --> Det Nominal

• Just need an attachment for
  – Det --> *a*
Revised

• So if the template for “every” is

\[ \{\lambda P.\lambda Q. \forall x P(x) \Rightarrow Q(x)\} \]

• What should the template for “a” be?
Recap

• Representing the meaning of declarative sentences using FOL

• From syntax to semantics
  – Rule-to-rule compositional approach
  – Requires lambda reduction

• Next time: on to lexical semantics!