Semantics

In order to understand language, we need to know its meaning.
- What is the meaning of a word? (Lexical semantics)
- What is the meaning of a sentence? ([Compositional] semantics)
- What is the meaning of a longer piece of text? (Discourse semantics)

NB: There are a lot of different approaches for each of these different aspects of semantics; since each approach typically focuses on one or two of these aspects (or particular phenomena within them), we don’t really have a “unified theory” of semantics.

Natural language conveys information about the world

We can compare statements about the world with the actual state of the world:
   Champaign is in California. (false)

We can learn new facts about the world from natural language statements:
   The earth turns around the sun.

We can answer questions about the world:
   Where can I eat Korean food on campus?

We draw inferences from natural language statements

Some inferences are purely linguistic:
   All blips are foos.
   Blop is a blip.
   Blop is a foo (whatever that is).

   John ate the cake.
   The cake was eaten by John.
We draw inferences from natural language statements

Some inferences require world knowledge.
Mozart was born in Salzburg.
Mozart was born in Vienna.
No, that can’t be — these are different cities.

Mozart was born in Salzburg.
Mozart was born in Austria.
Yes, that is correct — Salzburg is a city in Austria.

Today’s lecture

Our initial question:
What is the meaning of (declarative) sentences?
Declarative sentences: “John likes coffee”.
(We won’t deal with questions (“Who likes coffee?”) and imperative sentences (commands: “Drink up!”))

Follow-on question 1:
How can we represent the meaning of sentences?

Follow-on question 2:
How can we map a sentence to its meaning representation?

What we won’t discuss here

How do we represent world knowledge?
(e.g. that Salzburg and Vienna are cities, that Austria is a country, etc.)
There is a lot of work on knowledge representation and ontologies, including old research on “semantic networks”, and recent work on so-called knowledge graphs that is becoming increasingly important for NLP and other areas of AI.

What do nouns and verbs mean?

In the simplest case, an NP is just a name: John
Names refer to entities in the world.

Verbs define n-ary predicates: depending on the arguments they take (and the state of the world), the result can be true or false.
What do sentences mean?

Declarative sentences (statements) can be true or false, depending on the state of the world:

*John sleeps.*

In the simplest case, they consist of a verb and one or more noun phrase arguments.

**Principle of compositionality (Frege):** The meaning of an expression depends on the meaning of its parts and how they are put together.

---

Predicate logic expressions

**Terms:** refer to entities
- Variables: x, y, z
- Constants: John’, Urbana’
- Functions applied to terms (fatherOf(John’))

**Predicates:** refer to properties of, or relations between, entities
- tall’(x), eat’(x,y), …

**Formulas:** can be true or false
- Atomic formulas: predicates, applied to terms: tall’(John’)
- Complex formulas: constructed recursively via logical connectives and quantifiers

---

First-order predicate logic (FOL) as a meaning representation language

**Formulas**

- **Atomic** formulas are predicates, applied to terms: 
  \( book(x), eat(x, y) \)

- **Complex** formulas are constructed recursively by
  - *negation* \( (\neg): \neg book(John') \)
  - *connectives* \( (\land, \lor, \rightarrow): book(y) \land read(x, y) \)
    - conjunction (and): \( \phi \land \psi \)
    - disjunction (or): \( \phi \lor \psi \)
    - implication (if): \( \phi \rightarrow \psi \)
  - *quantifiers* \( (\forall x, \exists x) \)
    - universal (typically with implication) \( \forall x[\phi(x) \rightarrow \psi(x)] \)
    - existential (typically with conjunction) \( \exists x[\phi(x)], \exists x[\phi(x) \land \psi(x)] \)

Interpretation: formulas are either **true or false.**
The syntax of FOL expressions

Term \[\Rightarrow\] Constant | \[\Rightarrow\] Variable | \[\Rightarrow\] Function(Term,...,Term)

Formula \[\Rightarrow\] Predicate(Term, ...Term) | \[\Rightarrow\] ¬ Formula | \[\Rightarrow\] \(\forall\) Variable Formula | \[\Rightarrow\] \(\exists\) Variable Formula | Formula \& Formula | Formula \lor Formula | Formula \rightarrow Formula

Some examples

John is a student:
\[\text{student(john)}\]

All students take at least one class:
\[\forall x \text{ student}(x) \rightarrow \exists y(\text{class}(y) \land \text{takes}(x,y))\]

There is a class that all students take:
\[\exists y(\text{class}(y) \land \forall x (\text{student}(x) \rightarrow \text{takes}(x,y))\]

FOL is sufficient for many Natural Language inferences

All blips are foos.
\[\forall x \text{ blip}(x) \rightarrow \text{foo}(x)\]
Blop is a blip.
\[\text{blip(blop)}\]
Blop is a foo
\[\text{foo(blop)}\]

Some inferences require world knowledge.
Mozart was born in Salzburg.
\[\text{bornIn(Mozart, Salzburg)}\]
Mozart was born in Vienna.
\[\text{bornIn(Mozart, Vienna)}\]
No, that can't be-
\[\land \sim \text{bornIn(Mozart, Salzburg)}\]
these are different cities

Not all of natural language can be expressed in FOL:

Tense:
It was hot \textit{yesterday}.
I \textit{will} go to Chicago \textit{tomorrow}.

Aspect:
I \textit{am going} to Chicago.

Modals:
You can go to Chicago from here.
You must go to Chicago tomorrow.

Other kinds of quantifiers:
Most students hate 8:00am lectures.
λ-Expressions

We often use λ-expressions to construct complex logical formulas:

- \( \lambda x. \phi(\ldots x \ldots) \) is a function where \( x \) is a variable, and \( \phi \) some FOL expression.

- β-reduction (called λ-reduction in textbook):
  Apply \( \lambda x. \phi(\ldots x \ldots) \) to some argument \( a \):
  \( (\lambda x. \phi(\ldots x \ldots) a) \Rightarrow \phi(\ldots a \ldots) \)
  Replace all occurrences of \( x \) in \( \phi(\ldots x \ldots) \) with \( a \)

- n-ary functions contain embedded λ-expressions:
  \( \lambda x. \lambda y. \lambda z. \text{give}(x, y, z) \)

Using Combinatory Categorial Grammar (CCG) to map sentences to predicate logic

Function application

Combines a function \( X/Y \) or \( X/Y \) with its argument \( Y \) to yield the result \( X \):

- \( \text{NP} \rightarrow \text{S\NP} \)
- \( \text{NP} \rightarrow \text{S\NP} \)
- \( \text{NP} \rightarrow \text{S\NP} \)
- \( \text{NP} \rightarrow \text{S\NP} \)

Type-raising and composition

Type-raising: \( X \rightarrow T/(TX) \)

Turns an argument into a function.

- \( \text{NP} \rightarrow \text{S/(S\NP)} \) (subject)
- \( \text{NP} \rightarrow \text{S/(S\NP)/(S\NP/NP)} \) (object)

Harmonic composition: \( X/Y \ Y/Z \rightarrow X/Z \)

Composes two functions (complex categories)

- \( \text{NP} \rightarrow \text{S\NP/NP} \)
- \( \text{NP} \rightarrow \text{S\NP/NP} \)

Crossing function composition: \( X/Y \ Y/Z \rightarrow X/Z \)

Composes two functions (complex categories)
Type-raising and composition

Wh-movement (relative clause):

Right-node raising:

CCG semantics

Every syntactic constituent has a semantic interpretation:

Every lexical entry maps a word to a syntactic category and a corresponding semantic type:

Every combinatory rule has a syntactic and a semantic part:

Function application:  \( X/Y : \lambda x.f(x) \rightarrow X : f(a) \)
Function composition:  \( X/Y : \lambda x.f(x) \rightarrow X/Z : \lambda y.g(y) \rightarrow X/Z : \lambda z.f(\lambda y.g(y)).z \)
Type raising:  \( X : a \rightarrow T/(T/X) : \lambda f.f(a) \)

An example

An example with semantics
Quantifier scope ambiguity

“Every chef cooks a meal”

- **Interpretation A:**
  For every chef, there is a meal which he cooks.
  \[
  \forall x [\text{chef}(x) \rightarrow \exists y [\text{meal}(y) \land \text{cooks}(y, x)]]
  \]

- **Interpretation B:**
  There is some meal which every chef cooks.
  \[
  \exists y [\text{meal}(y) \land \forall x [\text{chef}(x) \rightarrow \text{cooks}(y, x)]]
  \]

**Interpretation A**

\[
\begin{align*}
\text{Every} & \quad \text{chef} & \quad \text{cooks} & \quad \text{a} & \quad \text{meal} \\
\frac{\lambda P \lambda Q \forall x [P x \rightarrow Q x]}{\lambda \exists . \text{chef}(z) / (S \backslash (S / \text{NP}) / \text{NP}) / \text{NP} / \text{NP}} & \quad \frac{\lambda u. \lambda v. \text{cooks}(u, v)}{\lambda \exists . \text{meal}(z) / (S \backslash (S / \text{NP}) / \text{NP}) / \text{NP}} & \quad \frac{\lambda \exists . \text{meal}(z)}{\lambda \exists . \text{meal}(z)} & \quad \frac{\lambda \exists . \text{meal}(z)}{\lambda \exists . \text{meal}(z)} \\
\frac{\lambda Q \forall x [\lambda z. \text{chef}(z) x \rightarrow Q x]}{\equiv \lambda Q \forall x [\lambda z. \text{meal}(z) y \land Q y]} & \quad \frac{\lambda Q \exists z [\lambda z. \text{meal}(z) y \land Q y]}{\equiv \lambda Q \exists z [\lambda z. \text{meal}(z) y \land Q y]}
\end{align*}
\]

**Interpretation B**

\[
\begin{align*}
\text{Every} & \quad \text{chef} & \quad \text{cooks} & \quad \text{a} & \quad \text{meal} \\
\frac{\lambda P \lambda Q \forall x [P x \rightarrow Q x]}{\lambda \exists . \text{meal}(z) / (S \backslash (S / \text{NP}) / \text{NP}) / \text{NP}} & \quad \frac{\lambda u. \lambda v. \text{cooks}(u, v)}{\lambda \exists . \text{meal}(z) / (S \backslash (S / \text{NP}) / \text{NP}) / \text{NP}} & \quad \frac{\lambda \exists . \text{meal}(z)}{\lambda \exists . \text{meal}(z)} & \quad \frac{\lambda \exists . \text{meal}(z)}{\lambda \exists . \text{meal}(z)} \\
\frac{\lambda w. \forall x [\text{chef}(x) \rightarrow \lambda u. \lambda v. \text{cooks}(u, v) w x]}{\equiv \lambda w. \forall x [\text{chef}(x) \rightarrow \text{cooks}(w, x)]} & \quad \frac{\lambda Q \exists z [\lambda z. \text{meal}(z) y \land Q y]}{\equiv \lambda Q \exists z [\lambda z. \text{meal}(z) y \land Q y]}
\end{align*}
\]

**Additional topics**

Representing events and temporal relations:
- Add event variables \(e\) to represent the events described by verbs, and temporal variables \(t\) to represent the time at which an event happens.

Other quantifiers:
- What about “most \(l\) at least two \(l\) … chefs”?

Underspecified representations:
- Which interpretation of “Every chef cooks a meal” is correct? This might depend on context. Let the parser generate an underspecified representation from which both readings can be computed.

Going beyond single sentences:
- How do we combine the interpretations of single sentences?
Today’s key concepts

Why do we need to represent meaning?
  Inference
  Interactions with (general) world knowledge and situational context

How do we represent meaning?
  First order predicate logic
  Semantics in CCG

Today’s reading

Textbook:
  Chapter 17, sections 1-3
  Chapter 18, section 2 (for a slightly different treatment of computational semantics)
  Optional: Chapter 18, section 3 (underspecified representations)