The Menu Bar

- Administrivia:
  - Lab 3b out; due April 12
  - Lab 4a on lexical semantics, out April 12

Agenda:
- What does this all mean?
- Frege’s principle of compositionality
- Representation and lambda calculus
Cognition as computation

- Computation manipulates formal symbols
- The symbols are represented
- The symbol manipulation is purely syntactic
- The symbol manipulation is semantically invariant

Our general view

- Syntactic representations to...
- Semantic representations to...
- Conceptual representations...
We know...

- What syntactic representations are
- We know much less about semantic or conceptual representations, but...
- Assume: they are the representations and vehicle for reasoning...
- So...must preserve what?
- Should be built up compositionally
- Why?

Compositionality, Turing, and all that

- Brown cow →
- Meaning(Brown) & Meaning(cow) & some mode of composition
- Why?

- Cf: Purple cow
Easy case

- Bob sleeps
- Bob likes ice-cream
- Event: $\text{likes}(\text{Bob, ice-cream})$

Hard case

(But the Accord was redesigned for the 2003 model year.)

The roomier, faster, and sleeker sedan’s sales stabilized last year, falling by just 1,230 units -- a strong showing in a market that saw combined total passenger car sales fall by 471,000 units.
The envelope please...

the(x1,e1&e3&e5&e7) & more'(e1,x1,y1,e2) & roomy'(e2,x1) & more'(e3,x1,y1,e4) & fast'(e4,x1) & more'(e5,x1,y1,e6) & sleek'(e6,x1) & sedan'(e7,x1) & poss(x1,z1) & sale(z1,x2) & Plur(z1,s1) & stabilize'(e8,s1) & Past(e8) & at-time(e8,y2) & last(y2,u1) & year(y2) & fall'(e9,s1) & by(e9,s2) & just(e6) & card'(e6,s2,1230) & unit(u2) & Plur(u2,s2) & Appos(e8,e11) & a(e11,e10&e11) & strong'(e10,e11) & show'(e11,x3,x4) & in(e10,m) & a(m,e12&e13) & market'(e12,m) & see'(e13,m,e14) & Past(e13) & combine(x5,s3) & total(s3) & passenger(p) & nn(p,c) & car(c) & nn(c,z2) & sale(z2,x6) & Plur(z2,s3) & fall'(e14,s3) & by(e14,s4) & card(s4,471000) & unit(u3) & Plur(u3,s4)

Why: recover meaning from structure – syntax-directed translation

[Diagram of syntactic tree]

S

VP(NP)=likes (Bob, ice-cream)

Bob

NP

VP= λy.likes(y, ice-cream)

λy λx likes(x, y)

likes

ice-cream
How: function application

What’s meaning? What’s semantics – 2 ends of the spectrum

- Answer 1: whatever it is, it’s mapping (translation) between representations
  And it depends on all of the text
- Answer 2: whatever it is, our answer depends on a much more focused task-specific question, viz., information extraction from texts
- Perhaps call this ‘natural language engineering’

- These two ends of the spectrum have different characteristics, and diff't uses
- Deep vs. Shallow?
What Counts as Understanding?

some notions

- We understand a statement if we know how to determine its truth.
  - What are the exact conditions under which it would be true?
    - necessary + sufficient
  - Equivalently, derive all its consequences
    - what else must be true if we accept the statement?
  - Philosophers tend to use this definition.
- We understand a statement if we can use it to answer questions.
  - Easy: John ate pizza. What was eaten by John?
  - Hard: White’s first move is P-Q4. Can Black checkmate?
  - Constructing a procedure to get the answer is enough.

Be able to translate

- Depends on target language
  - English to English? bah humbug!
  - English to French? reasonable
  - English to Chinese? requires deeper understanding
  - English to logic? deepest
    - all humans are mortal = ∀x [human(x) ⇒ mortal(x)]

- Assume we have logic-manipulating rules to tell us how to act, draw conclusions, answer questions ...
Answer 1: translation – from ‘syntactic’ rep to ‘semantic’ rep, aka “Deep”

- Mirrors the programming language approach
- **When** is it used?
- DB Q&A (but answer 2 can be used here... when and how?)
- Text understanding: when all the text is relevant - voice, inference, paraphrase, important
- Intentions, beliefs, desires (non-extensional = not just sets of items)

What requirements must meaning representations fulfill?

- **Verifiability:** The system should allow us to compare representations to facts in a Knowledge Base (KB)
  - Cat(Huey)
- **Ambiguity:** The system should allow us to represent meanings unambiguously
  - ‘German teachers’ has 2 representations
- **Vagueness:** The system should allow us to represent vagueness
  - He lives somewhere in the south of France.
Requirements: Canonical Form

- Inputs that mean the same thing have the same representation.
  - Huey eats kibble.
  - Kibble, Huey will eat.
  - What Huey eats is kibble.
  - It’s kibble that Huey eats.
- Alternatives
  - Four different semantic representations
  - Store all possible meaning representations in Knowledge Base

Requirements: Semantic Ambiguity

- Parallel to syntactic ambiguity
  - Happy [cats and dogs] live on the farm
  - [Happy cats] and dogs live on the farm
- Independent of syntactic structure
  - Every boy loves a dog
  - “all boys love a single dog”
  - “foreach boy, there is a dog he loves”
Requirements: Inference

- Draw valid conclusions based on the meaning representation of inputs and its store of background knowledge.
  Does Huey eat kibble?
  thing(kibble)
  Eat(Huey,x) ^ thing(x)

Word Senses & Ambiguity

- Q: Can the basic unit of meaning rep be a word?
- A: No, words have different *senses*
- Example: *go* has many senses (to move, depart, pass, vanish, reach, extend, ...)
- Senses are organized into an *ontology*
Requirements: Word Senses

- Ontology
  - Example: Aristotle’s classes
    - substance (physical objects)
    - quantity (e.g., numbers)
    - quality (e.g., being red)
    - Others: relation, place, time, position, state, action, affection
  - Important: actions, events
    - Provide a structure for organizing the interpretation of sentences

Requirements: Actions and Events

- *We lifted the box. It was hard work.*
  - The pronoun *it* refers to the whole action (not just the *box*)
- *We lifted the box. It was heavy.*
  - The pronoun *it* refers to the *box*
Need some kind of logical calculus

- Not ideal as a meaning representation and doesn't do everything we want - but close
  - Supports the determination of truth
  - Supports compositionality of meaning
  - Supports question-answering (via variables)
  - Supports inference
- What are its elements?
- What else do we need?

Logical Form Language

- Similar to first-order predicate calculus (FOPC)
  - Constants: word senses
  - Terms: constants that describe objects in the world
  - Predicates: constants that describe relations or properties
  - Propositions: predicate + terms
First order predicate calculus (FOPC)

Propositional logic: Don’t look inside propositions: P, Q, R, ...
First-order logic: Look inside propositions: p(x,y), like(J,M), ...

Constants: John1, Sam1, ..., Chair-46, ..., 0, 1, 2, ...
Variables: x, y, z, ....
**Predicate** symbols: p, q, r, ..., like, hate, ...
**Function** symbols: motherOf, sumOf, ...
All the logical connectives of propositional logic.

Predicates and functions apply to a fixed number of arguments:
- Predicates: like(John1, Mary1), hate(Mary1, George1), tall(Sue3), ...
- Functions: motherOf(Sam1) = Mary1, sumOf(2,3) = 5, ...

In the expression: 3 + 2 > 4
function predicate

Predicates applied to arguments are propositions and yield True or False.
Functions applied to arguments yield entities in the domain.

Predicates

- **Fido is a dog**
  (DOG1 FIDO1) unary predicate
- **Sue loves Jack**
  (LOVES SUE1 JACK1) or LOVES(Sue, Jack) binary predicate
- We shall place this into an event structure:
  Event(Loves1 :Agent Sue1 :Patient Jack1 :Time: present)
Extension of a predicate

The semantics of a unary predicate is the set of all entities in the domain for which the predicate is true.

The predicate dog → the set of all dogs (in the real world)

This is the extension of the predicate dog.

That leaves out possible dogs, future dogs, etc.; makes dog-ness depend on 'accidental' historically contingent properties of the world.

Possible Worlds

Possible world: A technical device in logic for handling "possible"

And a very powerful tool for analyzing some concepts.

You can use them without believing in them.

Duality between possible worlds and propositions:

A proposition can be viewed as the set of all possible worlds in which the proposition is true.

A possible world can be viewed as the set of all propositions that are true in it.

Add another proposition that has to be true

↔ Make the set of possible worlds smaller
Possible worlds to define ‘intension’ of a predicate

**Intension**: Map the predicate dog into a mapping from all possible worlds to the set of dogs in that possible world.

\[ \text{the predicate dog} \rightarrow [F: \text{possible world } w \rightarrow \text{the set of dogs in } w] \]

Given a predicate and a possible world, the intension will tell you the set of things that satisfy that predicate in that world.

Intension does a better job of capturing the essence of the concept.

---

A simple semantics for sentences

- Assuming that meaning of sentence is the proposition \( p \) expressed by sentence
- Simply its ‘truth conditional’ content, I.e,
  \[ p:w \rightarrow \{0,1\} \quad (w = 'a possible world') \]

This function (the proposition \( p \) expressed by \( s \)) may be viewed as:
- The truth conditions of a sentence \( s \)
- Assigning the values 0 or 1 for any given \( w \)
- Or as the set of possible worlds or situations where \( s \) is true
From syntactic structures to semantic structures

- We know what the structure of a simple subject-predicate sentence is.
- We also know its meaning: the proposition of set of (all possible, not just actual) situations given by \{sit | Peter sleeps in sit\}
- Or: where individual denoted by “Peter” is in the extension of the predicate sleeps, i.e., in the set of all individuals that sleep.

Syntax to semantics

\[
S \\
\rightarrow \text{SLEEP(Bob)}
\]

\[
\begin{array}{c}
S \\
NP \\
Bob \\
VP \\
sleeps
\end{array}
\]
The master principles

- Compositionality

- In a structure like this:
  \[ S, \; S^* \]
  \[ \begin{array}{c}
  NP, \; NP^* \; VP, \; VP^*
  \end{array} \]

  The meaning of the S is computed as the function application of the meaning of the VP to the meaning of the NP:
  \[ S^* = VP^*(NP^*) \]

- Intuitively: the concept expressed by the VP is asserted of the object to which the NP refers

The Principles

- Rule-to-rule hypothesis (Frege): semantic interpretation guided by syntactic structure;
  For each syntactic rule, there is a corresponding rule of semantic interpretation

- Compositionality

  We assume that the meaning of a complex expression is determined by the meaning of its parts
How to execute?

- Composition as function composition, I.e., function application
- We’ll need a way to express this...
- Also need a way to express predicates generally...

NP meanings

- If just a common noun (CN), e.g., “Bob”, “ice-cream”, then it’s like a constant (i.e., picks out all the “Bobs” in the world...)
- We’ll see how to express this in a moment...
VP meanings

- VP - sleeps (as intransitive)
- The meaning of the VP `sleeps`, then, is a function $f$ from an individual $x$ into a proposition (or a set of situations)
  $$f(x) = \{\text{situation} \mid x \text{ sleeps in situation}\}$$

How can we express this function?

6.001 to the rescue

- The function $f$ can be given by the $\lambda$-expression
  $$\lambda x \text{ SLEEPS}(x)$$
- When this function is applied to the argument ‘Bob’, as usual this binds the variable $x$:
  $$\lambda x \text{ SLEEPS}(x)\text{Bob} \Rightarrow \text{SLEEPS(Bob)}$$
Abstraction to the rescue

- SLEEPS(BOB) is composed of the VP meaning which is the function \( \lambda x \text{ SLEEPS}(x) \), applied to an argument, the NP meaning, which is Bob.

- Execution: associate with each context-free rule a corresponding semantic rule.

---

Context-free semantics

<table>
<thead>
<tr>
<th>Item or rule</th>
<th>Semantic translation</th>
</tr>
</thead>
<tbody>
<tr>
<td>( S \rightarrow \text{ NP } \text{ VP} )</td>
<td>( S^* : \text{ apply } \text{ VP}^<em>(\text{NP}^</em>) )</td>
</tr>
<tr>
<td>( \text{ VP } \rightarrow \text{ sleeps} )</td>
<td>( \text{ VP}^* : \lambda x \text{ SLEEPS}(x) )</td>
</tr>
<tr>
<td>( \text{ NP } \rightarrow \text{ CN} )</td>
<td>( \text{ NP}^* : \lambda x.x )</td>
</tr>
<tr>
<td>( \text{ CN } \rightarrow \text{ Bob} )</td>
<td>( \text{ CN}^* : \text{'Bob'} (\text{ie, a constant}) )</td>
</tr>
</tbody>
</table>
It all works...

S*: apply VP*(NP*)

\[ \lambda x \text{SLEEPS}(x) \quad \lambda x. x \quad 'Bob' \]

\[ \lambda x \text{SLEEPS}(x).\text{Bob} \]

SLEEPS(BOB)

OK, the next step... meaning of a transitive verb

- Bob likes ice-cream
- We already know the meaning of a VP likes *sleeps*, so we know the meaning of, e.g., ‘likes ice-cream’
- But what is the meaning of *likes*?
- \{situation | Bob likes ice-cream in situation \}
- We need a function that combines w/ ice-cream
- Goal: yield an intransitive VP meaning, as above,
- Intransitive: \( \lambda x \text{Likes-ice-cream}(x) \)
Transitive verb meaning

- Intransitive: \( \lambda x \) Likes-ice-cream(x)
- \( \lambda y \) g(y) \( \rightarrow \) LIKES(ice-cream)
- Lambda abstract:
  \( \lambda y \) LIKES(y) for the VP
- Replace this in Likes-ice-cream(x):
  \( \lambda x (\lambda y \) LIKES(x, y)) \) or to fix order
  \( \lambda y \lambda x \) LIKES(x, y).
  ice-cream. Bob
This is the meaning of likes

This gives us:

\[
S \quad \text{VP(NP)=likes (bob, ice-cream)}
\]

\[
\text{Bob} \downarrow \text{NP}
\]

\[
\text{VP= } \lambda y \text{likes(y, ice-cream)}
\]

\[
\text{V} \downarrow \text{likes}
\]

\[
\lambda y \lambda x \text{likes(x, y)}
\]

\[
\text{NP} \quad \text{ice-cream}
\]

\[
\text{Bob} \downarrow \text{likes}
\]

\[
\text{ice-cream}
\]
From sentence meanings to phrase meanings – intermediate summary

- Sentence meanings are propositions or sets of possible worlds or situations – those situations where the sentence is true
- NP meanings (meanings of proper names) are individuals
- Intransitive verb meanings are functions from individuals to sentence meanings (propositions)
- Transitive verb meanings are functions from individuals to intransitive verb meanings

Are we done?

- I wish!
- All the NPs so far are proper names, and so ‘constants’ – referring expressions
- Now we must consider lots more...
- The ice-cream, an ice-cream on the table, every ice-cream,...so much ice-cream, so little time...
- Bob likes no ice-cream..
The trouble with tribbles

- What would FOPC be for:

- Every person likes ice-cream

\[ \forall x \ (\text{Person } x \rightarrow \text{LIKE}(x, \text{ice-cream})) \]
\[ \exists x \ (\text{Person } x \& \text{LIKE}(x, \text{ice-cream})) \]

Let’s try our trick out on this...

Quantifiers cause problems

- If we apply composition following the syntax, what do we get?
- \[ \lambda y \lambda x \ \text{LIKES}(x, y). \text{ice-cream}, \forall x \ (\text{Person } x) \]
- But this yields:
  \[ \text{LIKES}(\forall x \ (\text{Person } x), \text{ice-cream}) \]
  \[ \forall x \ (\text{Person } x \rightarrow \text{LIKE}(x, \text{ice-cream}) \]

What happened to the NP ‘every person’?
What to do???
The solution next time...!

- But there is a lot more to do...