Review of Compositional Semantics

1 Review of Compositional Semantics

• Preliminary notions and concepts:
  – Truth conditions
    The sentence “______________” is true if and only if ____________.
  – Extension (of a sentence, 1-place predicate, 2-place predicate, ...)
    \([X]^w\) (‘the extension of \(X\) in \(w\)’)
  – The principle of compositionality

• Type theory
  – Types
    * Basic types: \(e\) for entities, \(t\) for truth values
    * Functional types: If \(\alpha\) and \(\beta\) are types, then \(\langle \alpha, \beta \rangle\) is a type.
  – Determine types of nodes in a tree:

<table>
<thead>
<tr>
<th>Syntactic category</th>
<th>Label</th>
<th>English expressions</th>
<th>Semantic type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sentence</td>
<td>S</td>
<td></td>
<td>(t)</td>
</tr>
<tr>
<td>Proper name</td>
<td>ProperN</td>
<td>John</td>
<td>(e)</td>
</tr>
<tr>
<td>e-type/referential NP</td>
<td>DP</td>
<td>the king</td>
<td>(e)</td>
</tr>
<tr>
<td>Common noun</td>
<td>CN</td>
<td>cat</td>
<td>(\langle e, t \rangle)</td>
</tr>
<tr>
<td>IV, VP</td>
<td>V(_{itr}), VP</td>
<td>run, love Kitty</td>
<td>(\langle e, t \rangle)</td>
</tr>
<tr>
<td>TV</td>
<td>V(_t)</td>
<td>love, buy</td>
<td>(\langle e, et \rangle)</td>
</tr>
<tr>
<td>Predicative ADJ</td>
<td>Adj</td>
<td>happy, gray</td>
<td>(\langle e, t \rangle)</td>
</tr>
<tr>
<td>Predicate modifier</td>
<td>Adj, Adv</td>
<td>skillful, quickly</td>
<td>(\langle et, et \rangle)</td>
</tr>
<tr>
<td>Sentential modifier</td>
<td></td>
<td>perhaps, not that</td>
<td>(\langle t, t \rangle)</td>
</tr>
<tr>
<td>Generalized quantifier</td>
<td>DP</td>
<td>someone, every cat</td>
<td>(\langle et, t \rangle)</td>
</tr>
<tr>
<td>Quantification</td>
<td></td>
<td>some, every, no, a</td>
<td>(\langle et, \langle et, t \rangle \rangle)</td>
</tr>
<tr>
<td>Determiner</td>
<td>D</td>
<td>the</td>
<td>(\langle et, e \rangle)</td>
</tr>
<tr>
<td>Definite determiner</td>
<td></td>
<td>who invited Andy</td>
<td>(\langle et, t \rangle)</td>
</tr>
<tr>
<td>Relative clause</td>
<td>REL</td>
<td></td>
<td>(\langle et, e \rangle)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>a</td>
<td>(\langle et, et \rangle, or \langle et, \langle et, t \rangle \rangle)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>is</td>
<td>(\langle et, et \rangle, or \langle e, et \rangle)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>that</td>
<td>(\langle t, t \rangle, or \langle et, e \rangle)</td>
</tr>
</tbody>
</table>
• Lambda calculus
  – Schema of lambda terms:
    \( \lambda v[\beta.\alpha] \) read as “the function which maps every \( v \) such that \( \beta \) to \( \alpha \)”
  – Lambda reduction/conversion
    \((\lambda v.\alpha)(x) = \alpha'\) where \( \alpha' \) is like \( \alpha \) but with every free occurrence of \( v \) replaced by \( x \).
  – Semantic types of lambda terms
    If \( v \) is of type \( \sigma \) and \( \alpha \) is of type \( \tau \), then \( \lambda v.\alpha \) is of type \( \sigma, \tau \).
  – Defining semantics of natural languages expressions using \( \lambda \)-notations
    * Predicates: \( \text{run}, \text{hit}, \text{cat}, \text{gray}, \text{larger than}, \text{from} \)
    * Other functions: \( \text{not}, \text{and}, \text{fast} \)
    * Vacuous words: \( \text{is}, \text{a}, \text{that} \)
      (Note that these words are usually semantically ambiguous)

• Semantic composition
  – Syntactic rules and Tree diagrams
    (Requirement: with provided phrase structure rules, draw a tree diagram for a sentence)
    * Phrase structure rules
      · Branching rules: \( A \rightarrow B \ C \)
      · Non-branching rules: \( A \rightarrow B \)
    * Vocabulary
  – Composition rules:
    * Basic rules: Terminal Nodes, Non-branching Nodes, Functional Application,
    * Other rules: Predicate Modification, Predicate Abstraction
  – Type mismatch

• Determiners and generalized quantifiers
  – Definite determiner: \( \text{the} \)
    * Uniqueness requirement of \( \text{the} \)
  – Quantificational determiner: \( \text{some}, \text{every}, \text{no} \)
    * Restrictor and scope of a quantificational determiner
  – Generalized quantifier: \( \text{someone}, \text{every cat} \)
    * Why is it that generalized quantifiers are not entities?

• Quantifier raising, movement, scope ambiguity
  – What motivates quantifier raising?
  – Quantifier raising is a covert movement taking place at the logical form.
  – How do you represent movement in compositional semantics?
2 Explaining the interesting facts!

- In the first week of this class, we saw a number of interesting semantic phenomena. Now let’s see how the concepts and technicalities learned in this class explain those phenomena.

- **Fact 1**: Sometimes, an inference implied by a positive sentence is also implied by the corresponding negative sentence:

  (1) a. Andy’s cooking is always bad.
     b. Andy’s cooking is not always bad.
     Both ab imply: Andy’s cooking is (at least) sometimes bad.

  (2) a. Suzi knows that Andy’s cooking is bad.
     b. Suzi doesn’t know that Andy’s cooking is bad.
     Both ab imply: Andy’s cooking is bad.

  Your explanation:

- **Fact 2**: Sometimes, the same sentence has multiple readings (*scope ambiguity of quantifiers*):

  (3) Every shark attacked a pirate.

  √ Every shark attacked a (different) pirate.  √ Every shark attacked the same pirate.

  Your explanation:
• **Fact 3:** Sometimes, a negative is not interpreted at where it is stated (**neg-raising**):

(4) John doesn’t believe that Mary won the race.

\[ = \text{John believes that Mary didn’t win the race.} \]

Explanations:


`believe` triggers a presupposition that the agent is opinionated about the truth/falsehood of the embedded clause. The assertion and this presupposition together entail the NR reading.

(5) John doesn’t believe \( p \).

\[ \text{not } [\text{John believes } p] \]

\[ \text{John believes } p, \text{ or John believes } \neg p \]

\[ \therefore \text{John believes } \neg p. \]


The unopinionated condition `John isn’t opinionated at \( p \)` is a stronger alternative of (??). Affirming the prejacent and denying this stronger alternative yield an NR reading.

(6) John doesn’t believe \( p \).

a. \( O \left[ \neg \text{John believes } p \right] \)

b. \( \text{ALT}(S) = \{ \neg [\text{John believes } p], \neg [\text{John believe } p \text{ or John believes } \neg p] \} \)

c. \( \neg [\text{John believes } p] \land \neg [\text{John believe } p \text{ or John believes } \neg p] \)

\[ = \neg [\text{John believes } p] \land [\text{John believe } p \text{ or John believes } \neg p] \]

\[ = \text{John believes } \neg p \]

• **Fact 4:** Semantics interacts prosody.

(7) a. We only asked ANDY to hand in homework one.

\[ \rightarrow \text{We didn’t ask Billy to hand in homework one.} \]

b. We only asked Andy to hand in homework ONE.

\[ \rightarrow \text{We didn’t ask Andy to hand in homework two.} \]


The stressed item is focused and is associated with a set of focus-alternatives (just like that a scalar item is associated with a set of scalar alternatives). `Only` presupposes the truth of its prejacent, and negates the focus alternatives that are not entailed by the prejacent.

(8) a. only \( S \text{ we asked ANDY to hand in homework one} \)

b. Alt (S) = {we asked \( x \) to hand in homework one: \( x \in \{\text{Andy, Billy}\} \})

c. \( \neg [\text{we ask Billy to hand in homework one}] \)

(9) a. only \( S \text{ we asked Andy to hand in homework ONE} \)

b. Alt (S) = {we asked Andy to hand in homework \( x \): \( x \in \{\text{one, two}\} \})

c. \( \neg [\text{we ask Andy to hand in homework two}] \)