A Review of the Essential Concepts of Extensional Semantics

These review notes will both (i) ensure that we’re all on the same page regarding certain foundational ideas and formalisms, and (ii) present those ideas in a way that will nicely set up the introduction of crucial new ideas in the following week.

1. The Big Picture

1.1 Our Ultimate Goal
A precise, formal theory of a particular sub-component the human language faculty: the ability to productively interpret the (infinite) sentences of our language(s).

How do we do this?
Since brains/lifetimes are finite, this knowledge must be represented in our brains as some kind of combinatoric system, one that comprises:

a. Finite number of primitive meaningful units (lexemes, lexical items)
b. Finite set of ‘rules’ for deriving the meaning of a complex expression from the meanings of the primitives and the structure of the complex expression

1.2 The Principle of Compositionality (Gottlob Frege, 1884)
The meaning of a complex expression (in natural language) can be effectively computed from (i) the meaning of its component expressions, and (ii) their ‘mode of combination’ (i.e., the syntax of the expression)

1.3 The Over-Arching Research Goal (for Semanticists)
Develop and test a theory of the rule system that our language faculty employs to compute the meanings of complex expressions from (i) the meanings of their component parts, and (ii) their syntactic structure (à la the Principle of Compositionality)

1.4 Some Related, More Specific Research Questions
a. How does a human being acquire this combinatoric system? How much is already specified by the biology of the organism? (Semantic Acquisition)
b. How does this combinatoric system vary across languages? Do languages differ in how they ‘compute meanings’, and if so, in what ways? (Semantic Typology)

---

2. **The Meaning of ‘Meaning’**

(5) **An Immediate Problem for Our Project: What is a ‘Meaning’?**

- To answer the question in (3), we want to develop some hypotheses about what the system is like, and then test them…
- So, we’ll want to design a hypothetical formal system that will manipulate primitive ‘meanings’ to derive the ‘meanings’ of more complex sentences.
- But, how do we formally represent the ‘meaning’ of a sentence or its component phrases?
  - What is the ‘meaning’ of “Dave” and “smokes” such that ‘combining them together’ gives us the ‘meaning’ of “Dave smokes”?

**Over-Arching Problem:**
- The word ‘meaning’ is a vague, pre-theoretic term from every-day discourse.
- Thus, it may not be an appropriate term for a precise, scientific study of human language.

(6) **New, Preliminary Objective**
Let’s replace our everyday, pre-theoretic concept of ‘meaning’ with something a bit more precise – narrower – which a formal system could perhaps derive/manipulate…

**What We Need to Do Now:**
Let’s try to pin down the phenomena that we’re really interested in, among those that are typically, loosely categorized under the general umbrella of ‘meaning’…

(7) **‘Meaning’ is as ‘Meaning’ Does**
“In order to say what a meaning is, we may first ask what a meaning does, and then find something that does that” (David Lewis; “General Semantics”)

  - What kinds of things do we know when we know ‘the meaning’ of an expression?

a. **Social Appropriateness of the Expression:** ‘fèces’ vs. ‘shit’

b. **The ‘Subjective Perspective’ of the Speaker:** ‘come’ vs. ‘go’

c. **The Informational Content of the Utterance:**
   What information about the world the speaker’s utterance ‘conveys’.

Rightly or wrongly, (7c) has received by-far-and-away the greatest attention over the centuries
It will also be the aspect of meaning that we will be concerned with in this course.
And so, let’s try to develop our concept of the ‘information’ that is ‘conveyed’ by a sentence...
2.1 The Different Ways that Information Can Be ‘Conveyed’

When we examine this notion of a sentence ‘conveying information’, we find that it’s not so simple either: *There are different ways that information can be ‘conveyed’ by a sentence.*

(8) **Example Dialog**

Person 1: How did Dave’s physical go?  
Person 2: Well, he’s stopped smoking.

*Person 2’s utterance ‘conveys’ all the following information:*
- Dave has stopped smoking.
- Dave *has been smoking*.
- Dave’s physical did not go well. (Dave received bad news at his physical.)

*Each of these different bits of information is ‘conveyed’ in a different way by the utterance.*

(9) **Assertion:** Information that is ‘explicitly added’ by the utterance.

The information that *Dave has stopped smoking* is **asserted** by the utterance / speaker

- **Diagnostic:** Sentence S **asserts** that \( p = S \) is true if and only if \( p \)

  **Test:** “Dave stopped smoking” is true if and only if Dave stopped smoking.

(10) **Presupposition:** Information that an utterance ‘takes for granted’, which has to hold for the utterance to even be meaningful…

The information that *Dave has been smoking* is **presupposed** by the utterance / speaker

- **Diagnostic:** Sentence S **presupposes** \( p = S \) is true or false only if \( p \)

  **Test:** “Dave stopped smoking” can only be true if Dave has been smoking.  
  “Dave didn’t stop smoking” can only be true if Dave has been smoking.

(11) **Implicature** Information that isn’t *explicitly* added by the utterance, but which the speaker (clearly) intends for the addressee to conclude.

The info that *D’s physical didn’t go well* is an **implicature** of the utterance / speaker

- **Diagnostic:** \( p \) is an **implicature** of \( S = \) \( p \) is ‘conveyed’ by \( S \), but ‘not \( p \)’ is consistent with \( S \)

  **Test:** “Dave stopped smoking, but he did fine on his physical” is logically consistent.
The Main Point:
For a full theory of how a sentence ‘conveys’ information, we need to understand:

(i) How the assertions of a sentence S are derived from the meanings of its parts.
(ii) How the presuppositions of S are derived from the meanings of its parts.
(iii) How the implicatures of S are derived (in part) from the meanings of its parts.

Ultimately, a formal semantic theory will need to do all of (i) – (iii) above.
However, we also need to start somewhere, and so – for better or worse – we will start with (i)…

(12) The Over-Arching Research Goal for Semanticists [RESTATED]

Develop and test a theory of the rule system that our language faculty employs to compute the asserted content of (declarative) sentences from (i) the meanings of their component parts, and (ii) their syntactic structure.

Side-Note: What about non-declarative sentences, like questions and imperatives? They seem to also be meaningful, but they don’t seem to ‘assert’ anything!...

Suspend your disbelief!
If all goes according to plan, we’ll get back to questions and imperatives later in the course…

2.2 The Importance of ‘Truth Conditions’ to a Theory of Meaning

(13) Special Terminology: Truth Conditions

The ‘truth conditions’ of a sentence S are the conditions under which S is true.

Truth-Conditional Statement: ‘S is true if and only if p’

Some Consequences:

a. The ‘truth conditions’ of S are another name for the ‘assertions’ of S
b. Thus, our goal in (12) can again be restated to the following:

(14) The Over-Arching Research Goal for Semanticists [RESTATED AGAIN]

Develop and test a theory of the rule system that our language faculty employs to compute the truth-conditions (of a declarative sentence) from (i) the meanings of their component parts, and (ii) their syntactic structure.
(15) **Another Way of Seeing the Relationship Between Truth-Conditions and ‘Meaning’**

- So far, we’ve seen that ‘truth-conditions’ are simply one way of recasting ‘asserted content’ (one important dimension of the overall meaning of a declarative sentence).

- But, we can also see that – when aided by a little pragmatics – computing the truth-conditions of a sentence allows a speaker to deduce information about the world!

- As in (16), if a listener assumes that a speaker is knowledgeable and truthful, then knowing the truth-conditions of their utterance informs them of the state of the world.

(16) **A Model of Information Computed During Sentence Comprehension**

a. **Speaker’s Utterance:** /ðə haʊs ɪz ən ˈfæʃ/ 

b. **Listener’s Computations:**

(i) **Syntax:** The string /ðə haʊs ɪz ən fæʃ/ has the following structure:
[[the house][is[on[fire]]]]

(ii) **Semantics:** [[the house][is[on[fire]]]] is true iff the house is on fire

(iii) **Pragmatics:** The speaker is an honest guy, so he believes what he says... The speaker is smart, so what he believes is true... So “[[the house][is[on[fire]]]]” must be true...
So, **given its truth conditions**, the house must be on fire...
...OMG THE HOUSE IS ON FIRE!!....

(17) **Brief Terminological Aside: ‘Object Language’ and ‘Meta-Language’**

a. **The Object Language:**
The language we are *describing* and/or *studying* (not necessarily using).

b. **The Metalanguage:**
The language we are *using* to characterize the truth conditions of the sentences of our object language.

- The object language and meta-language can both be *the same language*, or they can be different languages!

“*The house is on fire*” is T iff the house is on fire. (OL & ML = English)

“*Wé hit yei kanagán*” is T (true) iff the house is on fire. (OL = Tlingit)
3. Obtaining a Theory of Truth Conditions from ‘Extensions’

The following is a crude sketch of our over-arching goal in (14):

\[
\text{(18) Crude Sketch of a Compositional Semantic Theory that Yields Truth Conditions}
\]

\[
\text{MEANING( [NP Barack ] ) } \oplus \text{ MEANING( [VP smokes ] ) } =
\]

\[
\text{MEANING( [s Barack smokes ] ) } =
\]

\[
\text{TRUTH-CONDITIONS( [s Barack smokes ] )}
\]

- So far, we’ve sharpened our notion of what we want to derive as the ‘meaning’ of the sentence ‘Barack smokes’ (i.e., its truth-conditions)…

- But now, what are the ‘meanings’ of ‘Barack’ and ‘smokes’ such that combining them together can yield for us the truth conditions of ‘Barack smokes’

3.1 More about the Meaning of ‘Meaning’: An Excursus of ‘Extensions’

We’ve already seen that the everyday word ‘meaning’ is vague and ambiguous in a number of ways…here’s another:

\[
\text{(19) The Meaning of the Phrase ‘The President’}
\]

a. In one sense, the meaning of the NP “the president of the United States” underwent a change last year. It went from meaning Donald Trump to meaning Joe Biden.

   (‘meaning’ = ‘denotation’, ‘reference’)

b. In another sense, the meaning of the NP is the same now as it was two years ago. To this day, it still means ‘the chief executive officer of the US government’.

   (‘meaning’ = ‘sense’, ‘concept’)

Instead of using the word ‘meaning’ in this vague and ambiguous fashion, let’s introduce two technical terms to refer unambiguously to these two different ‘senses’ of the word “means”.
(20) **Extension vs. Intension (of NPs)**

a. **Extension of an NP** *(a.k.a., Reference / Denotation of an NP)*
   - Thing in the world the NP (currently) refers to / picks out / denotes / designates
   - The *extension* of “the president” is *Joe Biden*.

b. **Intension of an NP** *(a.k.a., Sense of an NP)*
   - ‘General concept’ behind the NP, *which determines* *(for a given time/situation)*
   - what the *extension* of the NP is…
   - The *intension* of “the president” is *the chief executive officer of the US*.

<table>
<thead>
<tr>
<th>Extension of ‘the president’</th>
<th>January 19, 2021</th>
<th>January 19, 2020</th>
</tr>
</thead>
<tbody>
<tr>
<td>Joe Biden</td>
<td>Donald Trump</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Intension of ‘the president’</th>
<th>January 19, 2021</th>
<th>January 19, 2020</th>
</tr>
</thead>
<tbody>
<tr>
<td>The chief executive officer of the government of the US</td>
<td>The chief executive officer of the government of the US</td>
<td></td>
</tr>
</tbody>
</table>

So, the ‘meaning’ of an NP can be broken up into its ‘extension’ and its ‘intension’…
- This basic fact had been recognized by philosophers going back centuries…
- But in the 1800s, Gottlob Frege asked a somewhat weird question:
  …*can the meaning of a sentence likewise be broken up in this way?*

(21) **Intension of a Sentence = ‘Truth Conditions’**

- We might take the ‘intension’ of a sentence to be (something like) its *truth conditions*…
- As we’ve already seen, the truth conditions of a sentence are akin to what we might vaguely describe at the sentence’s (asserted) ‘informational content’.

... *But if the ‘intension’ of a sentence is its truth conditions, what is its ‘extension’?...*

(22) **Extension of a Sentence = Truth Value**

- If we take the ‘intension’ of a sentence to be its truth conditions, then we should take the ‘extension’ of a sentence to be its *truth value*.
- **Why?**
  - Recall that the ‘intension’ determines for a given time/situation what the *extension* is.
  - *Truth conditions* determine for a given time/situation what the *truth value* is.
The General Picture

a. Intension of “X”:
A kind of ‘concept’/ ‘definition’ which – for any given time/situation – determines what “X” ‘picks out’ at that time/situation.

b. Extension of “X”:
The thing which, at a given time/situation “X” ‘picks out’

c. Illustration:

<table>
<thead>
<tr>
<th>Extension of ‘the president’</th>
<th>January 19, 2021</th>
<th>January 19, 2020</th>
</tr>
</thead>
<tbody>
<tr>
<td>Joe Biden</td>
<td>Donald Trump</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Intension of ‘the president’</th>
<th>January 19, 2021</th>
<th>January 19, 2020</th>
</tr>
</thead>
<tbody>
<tr>
<td>The chief executive officer of the government of the US</td>
<td>The chief executive officer of the government of the US</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Extension of ‘The president has a dog’</th>
<th>January 19, 2021</th>
<th>January 19, 2020</th>
</tr>
</thead>
<tbody>
<tr>
<td>TRUE</td>
<td>FALSE</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Intension of ‘The president has a dog’</th>
<th>January 19, 2021</th>
<th>January 19, 2020</th>
</tr>
</thead>
<tbody>
<tr>
<td>Truth conditions of ‘The president has a dog’</td>
<td>Truth conditions of ‘The president has a dog’</td>
<td></td>
</tr>
</tbody>
</table>

What’s the point of all of this?

*As we will presently see, you can actually build a decent theory of how meanings can ‘compose’ to yield truth-conditions by paying attention to extensions (defined as above)!*

3.2 Towards a Compositional Semantics Based on Extensions

(24) Some New Notation  

\[
[[ X ]] = \text{the extension of “X”}
\]

(25) The Compositionality of Extensions

a. We’ve broken down the ‘meaning’ of a S/NP into its extension and its intension.

b. Recall, however, that our semantic system must be such that the ‘meaning’ of a complex expression should be derived from the ‘meaning’ of its component parts.

c. Thus, the extension of a complex expression should be derived from the extensions of its component parts!

d. In A Picture:

\[
[[ \text{Barack} ]] \oplus [[ \text{smokes} ]] = [[ \text{Barack smokes} ]] = \text{True}
\]

\[
[[ \text{Rajesh} ]] \oplus [[ \text{smokes} ]] = [[ \text{Rajesh smokes} ]] = \text{False}
\]
(26)  **The Extension of a Predicate = Function**

a. In order to make the picture in (9d) work, the extension of the predicate ‘smokes’ must ‘combine’ with the extension of ‘Barack’ to yield the extension of ‘Barack smokes’, which is the value TRUE…

\[\text{... and it must combine with the extension of ‘Rajesh’ to yield the extension of ‘Rajesh smokes’, which is FALSE}\]

b. **How can we model this?** Well, we know the following:

\[
\begin{align*}
(i) & \quad [[Barack]] = \text{Barack} & (iii) & \quad [[Rajesh]] = \text{Rajesh} \\
(ii) & \quad [[Barack smokes]] = \text{T(true)} & (iv) & \quad [[Rajesh smokes]] = \text{F(false)}
\end{align*}
\]

c. **Thus:** \[ [[\text{smokes}}] \oplus \text{Barack} = \text{T} \quad \text{and} \quad [[\text{smokes}}] \oplus \text{Rajesh} = \text{F}\]

d. **Key Pattern:** If any entity x is a smoker, then \[ [[\text{smokes}}] \oplus x = \text{T}\]
If any entity x is a non-smoker, then \[ [[\text{smokes}}] \oplus x = \text{F}\]

e. **The Formal Intuition:** *So, the extension of ‘smokes’ is like a FUNCTION! It takes an entity x as input, and outputs T iff x smokes!*

f. **The Core Idea:** \[ [[\text{smokes}}] = [\lambda x : x \text{smokes} ]\]

(27)  **Interim Summary: Some Illustrative Extensions**

\[
\begin{align*}
a. & \quad [[\text{Barack}}] = \text{Barack} \\
b. & \quad [[\text{Rajesh}}] = \text{Rajesh} \\
c. & \quad [[\text{smokes}}] = [\lambda x : x \text{smokes} ]
\end{align*}
\]

We’ve almost got the picture in (25) worked out… all we need is a rule for ‘combining’ the extensions of ‘Barack’ and ‘smokes’ to yield the extension of ‘Barack smokes’

(28)  **The Rule of Function Application [FA]**

If X is a branching node that has two daughters – Y and Z – and if \([[[Y]]]\) is a function whose domain contains \([[[Z]]]\), then \([[X]] = [[[Y]]][[[Z]]])

With this rule, we now have a system that derives the extension of the sentence ‘Barack smokes’ from (i) the extension of its component pieces, and (ii) the syntax of the sentence.
(29) **Computing the Extension of ‘Barack smokes’**

a. **Simplified Syntactic Assumption:**
The structure of the sentence *Barack smokes* is as follows:

```
\[ S \]
```

```
Barack \hspace{1cm} \text{smokes}
```

b. **Semantic Derivation:**

(i) \[
[[ S ]] = \text{(by F(unction) A(pplication))}
\]

(ii) \[
[[ \text{smokes} ]] ([[ \text{Barack} ]]) = \text{(by (27a))}
\]

(iii) \[
[[ \text{smokes} ]] (\text{Barack}) = \text{(by (27c))}
\]

(iv) \[
[ \lambda x : x \text{smokes} ] (\text{Barack}) = \text{(by the facts of the world)}
\]

(v) \[
\text{True}
\]

*So, the system in (27) and (28) can derive the extension of a sentence (its truth value) from the extension of its component parts (given the facts of the world)…*

So, we’ve obtained the system sketched in (25)…

...Ok, but so what?...

---

3.3 **From Extensions to Truth Conditions**

(30) **Our Desired Semantic System**

A precise, formal system that for every sentence $S$ of the language (*i.e.*, English), will derive a correct statement of the following form: “$S$ is True iff $X$”

Believe it or not, our system for deriving extensions (partly) achieves what we want in (30)!!!

**Side-Note:** Some logical truths to keep in mind

| a. | Transitivity of ‘iff’ | A $B$ iff $B$ and $B$ iff $C$ entails $A$ iff $C$ |
| b. | Substituting ‘equals’ for ‘equals’ | If $x = y$, then $x = z$ iff $y = z$ |
Deriving the Truth Conditions of a Sentence via Our Theory of Extensions

To Prove: "S" \(\text{iff}\) Barack smokes

a. "S" \(\text{iff}\) (by definition of our notation)

b. [[S]] = T \(\text{iff}\) (by FA)

c. [[smokes]]([[Barack]]) = T \(\text{iff}\) (by (27))

d. \(\lambda x : x\text{smokes}\)(Barack) = T \(\text{iff}\) (by definition of our \(\lambda\)-notation)

e. Barack smokes

Putting it all together: (31a) \(\text{iff}\) (31e) ; [[Barack smokes]] = T \(\text{iff}\) Barack smokes

What just happened:

- In the preceding section, we first showed how a compositional extensional semantics can, given the facts in the world, compute the truth value of a sentence.

- Conversely, if we take as hypothesis that the truth value of a sentence is ‘TRUE’, our compositional extension semantics can ‘work backwards’, and compute how the world must be constituted in order for the sentence to be true!

- Thus, we can use our extensional semantic function “[[ ]]” to compute the truth conditions of sentences!!!

The Big Upshot

An ‘extensional semantics’ – a formal system that maps complex structures to their extensions in the world – can provide us with a theory of how our brains recursively map sentences to their truth conditions.
(33) **Our Over-Arching Project, Redefined Again**

We wish to develop the *right* theory of the function “[[ . ]]”, by examining the truth-conditions of particular sentences of the language. Such a theory will consist of:

a. Primitive statements for the lexical items of the language

b. Rules for deriving the value of “[[ ]]” for larger structures from (i) the value of “[[ ]]” for their component parts and (ii) the syntax of the larger structures

So our task is to adjust (a) and (b) until our theory predicts *exactly* the correct truth conditions for every sentence of English!

- Such a theory will provide a model of how our brains map expressions of our language to their truth conditions…
  - Which is *one small part* of understanding how our brains map expressions of our language onto their ‘meanings’…

4. **On Devising Semantic Hypotheses for Lexical Items**

In Section 3.2, we developed a theory of the semantic value of the intransitive verb ‘smokes’, whereby we identified its extension with a particular type of function (from entities to T-values).

...*It’s instructive to reflect on how we came to this conclusion:*

(34) **Determining the ‘Meaning’ of a Lexical Item L**

a. Consider the truth-conditions of sentences in which L appears.

b. Consider the (already established) extensions of the other lexical items in these sentences.

c. Based on (a) and (b), develop an entry for L which would – in combination with the entries for the other words in the sentences (b) – correctly derive the truth conditions of the sentences it appears in (a).

Thus, we base our semantics for lexical items only on how those items contribute to the truth-conditions of a sentence…

- As you saw in LING 610, this kind of an approach works especially well for the semantic analysis of *function words* like ‘*and*’, ‘*the*’, ‘*every*’, *etc.*

- For *content words* like ‘*smokes*’ and ‘*child*’, however, this approach admittedly ignores (or can ignore) much of their intuitive ‘meaning’ *(e.g., the entry in (27b) doesn’t explicitly say anything about what distinguishes ‘smoking’ from ‘drinking’…)*
5. The System of Semantic Types

The extensional semantics developed in Section 3 includes some assumptions about what types of things can be the extensions of natural language expressions:

- Entities (e.g. Barack)
- Truth Values (e.g. True, False)
- Functions from Entities to Truth-Values (e.g. \[ \lambda x : x \text{smokes} \])

As you saw in LING 610, semanticists find it useful to have a somewhat formal / rigorous way of talking about the types of semantic values that natural language expressions can have.

(35) Terminology: ‘Semantic Type’
A ‘type of thing’ that natural language expressions can have as their semantic value.

(36) The Theory of Semantic Types, Part 1: Basic Types
a. \(e\) is a semantic type \((e = \text{‘entity’})\)

b. \(t\) is a semantic type \((t = \text{‘truth value’})\)

(37) The Theory of Semantic Types, Part 2: Functional Types
If \(\alpha\) is a semantic type, and \(\beta\) is a semantic type, then \(<\alpha,\beta>\) is a semantic type. \((<\alpha,\beta> = \text{‘function from } \alpha \text{ to } \beta’)\)

(38) Illustration of Some Semantic Types
- \(e\) (entities)
- \(t\) (truth-values)
- \(<e,t>\) (functions from entities to truth values)
- \(<e,<e,t>>\) (functions from entities to functions form entities to truth-values)
- \(<<et>,<et,t>>\) (functions from \([\text{functions from entities to truth-values}]\) to \([\text{functions from entities to truth-values}]\) to truth values)

(39) Notations for Domains
If \(\alpha\) is a semantic type, then \(D_\alpha\) is the set of all things \((a.k.a. \text{‘the domain’})\) of type \(\alpha\)
- \(D_e\) = the set of entities
- \(D_{<e>}\) = the set of functions from entities to truth-values
- \(D_{<e,t>}\) = the set of functions from \([\text{functions from entities to truth-values}]\) to \([\text{functions from entities to truth values}]\)
6. The Lambda Notation for Functions

In Section 3, we represented [[smokes]] as ‘[λx : x smokes]’. Let’s briefly review how this λ-notation works to define functions.

(40) Lambda Notation, Part 1

a. Syntax: [λx : x ∈ D . φ(x)]

b. Semantics: The function whose domain is D (i.e., which takes as argument anything in the set D), and for all x ∈ D, maps x to φ(x)

(41) Examples:

a. [λx : x ∈ {0, 1, 2, 3} . x + 3] = {<0,3>, <1,4>, <2,5>, <3,6>}

b. [λx : x ∈ {Seth, Rajesh} . the office of x] = {<Seth, 426>, <Rajesh, 418>}

(42) Lambda Notation: Functions Taking Arguments

[λx : x ∈ D . φ(x)](a) = the unique y such that <a,y> ∈ [λx : x ∈ D . φ(x)]

= the function ‘[λx : x ∈ D . φ(x)]’ taking a as argument

(43) Examples

a. [λx : x ∈ {0, 1, 2, 3} . x + 3](2) = 5

b. [λx : x ∈ {Seth, Rajesh} . the office of x](Rajesh) = 418

(44) The Rule of ‘Lambda Conversion’ [LC]

- The following equation is a straightforward consequence of how our notation is defined…
- But, since we’ll be using it quite a bit, it’s nice to have a name for it: ‘Lambda Conversion’ (LC)

[λx : x ∈ D . φ(x)](a) = φ(a)

(45) Examples

a. [λx : x ∈ {0, 1, 2, 3} . x + 3](2) = (by LC) 2 + 3 = 5

b. [λx : x ∈ {Seth, Rajesh} . the office of x](Rajesh) = (by LC)

the office of Rajesh = 418
The True Power of This Notation

- The real advantage of lambda notation is that it offers a very handy and simple way of defining functions that yield other functions as values.
- The way to represent such functions is incredibly simple: You just embed one lambda formula inside another one!

Example

\[ \lambda x : x \in N \ . \ [ \lambda z : z \in N . x + z ] \]

The function which takes a number \( x \) as argument and returns the function which takes a number \( z \) as argument and returns \( x + z \)

Crucial Question / Problem

- How do we represent a function like \( [[ \text{smokes} ]] \) using \( \lambda \)-notation?
- Note that this function takes an argument and returns a particular value (i.e., \( T \)) only if some condition is met (i.e., if that argument smokes)

Lambda Notation, Part 2

a. Syntax: \[ \lambda x : x \in D \ . \ \text{IF } \varphi(x) \ \text{THEN} \ y, \ \text{ELSE} \ z \]

b. Semantics: The function whose domain is \( D \) (i.e., which takes as argument anything in the set \( D \)), and for all \( x \in D \),
   - maps \( x \) to \( y \) if \( \varphi(x) \),
   - and maps \( x \) to \( z \) otherwise

Example:

\[ \lambda x : x \in D_c \ . \ \text{IF} \ x \text{smokes} \ \text{THEN} \ T, \ \text{ELSE} \ F \]

a. The function whose domain is \( D_c \), and for all \( x \in D_c \), maps \( x \) to \( T \) iff \( x \text{ smokes} \)

b. \( [[ \text{smokes} ]] \)

---

2 To save space, I will write ‘\( N \)’ for the set \{ \( x : x \) is a whole number greater than 0 \}
3 This notation is not found in Heim & Kratzer (1998). A highly technical discussion of it can be found at the following: http://en.wikipedia.org/wiki/Lambda_calculus#Logic_and_predicates
Another Example

\[ [ \lambda y : y \in D_e . [ \lambda x : x \in D_e . \text{IF } x \text{ likes } y \text{ THEN } T, \text{ELSE } F ] ] \]

a. The function whose domain is \( D_e \) and for all \( y \in D_e \), maps \( y \) to…
   the function whose domain is \( D_e \), and for all \( x \in D_e \), maps \( x \) to \( T \) iff \( x \) likes \( y \)

b. \([[[ \text{likes }]]]\)

Some Abbreviations Regarding the Domain of the Function

Sometimes, to save space, we might abbreviate the statement of the function’s domain in the following ways:

\[ [ \lambda y : y \in D_x . \ldots ] = \begin{cases} (i) & [ \lambda y \in D_x : \ldots ] \\ (ii) & [ \lambda y x : \ldots ] \\ (iii) & \text{when it’s clear from context what the domain of the function is, we can even just write: } [ \lambda y : \ldots ] \end{cases} \]

A (Potentially Confusing) Abbreviation for Functions Mapping to Truth-Values

- Note that, given our definition in (40), if ‘\( \varphi(x) \)’ is a meta-language sentence, then it isn’t immediately clear what ‘\( [ \lambda x : x \in D . \varphi(x) ] \)’ should mean.

  Example: \[ [ \lambda x : x \in D_e . \text{x smokes} ] = [ \lambda x : x \in D_e . \text{IF } \text{x smokes } \text{THEN } T, \text{ELSE } F ] \]
  ‘the function that maps an entity \( x \) onto… \text{x smokes}?!?
  …\text{what the heck is ‘}\text{x smokes}\text{’?}’

- Consequently, we can use such structures as (unambiguous) abbreviations of the notation in (49)-(51)!

  a. The Abbreviatory Convention:
  If ‘\( \varphi(x) \)’ is a meta-language sentence then ‘\( [ \lambda x : x \in D . \varphi(x) ] \)’ is short-hand for ‘\( [ \lambda x : x \in D . \text{IF } \varphi(x) \text{ THEN } T, \text{ELSE } F ] \)’

  b. Example: \[ [ \lambda x : x \in D_e . \text{x smokes} ] = [ \lambda x : x \in D_e . \text{IF } \text{x smokes } \text{THEN } T, \text{ELSE } F ] = \]
  ‘The function that maps an entity \( x \) to \( T \) iff \( x \) smokes’

**NOTE:**

- The abbreviatory convention in (53) is not part of standard \( \lambda \)-notation.
- However, since its introduction in Heim & Kratzer (1998), it has become rather widespread throughout natural language semantics.
7. **Interim Summary**

Thus far, the key concepts and formalisms reviewed here are sufficient for a basic (introductory) treatment of the following kinds of phenomena:

- Transitive and ditransitive verbs
- Logical connectives (*and, or, not*)
- Adjectival modification
- Definite descriptions (and other presuppositional triggers)
- Quantificational NPs (DPs)

Consequently, these notions form the backbone of the first half (or so) of LING 610… Students are advised to spend some time reviewing their notes on these topics from LING 610…

- However, even within extensional semantics, more advanced topics rely upon two additional devices: (i) pronominal indices, and (ii) the rule of Predicate Abstraction

- **Thus, in the following introductory handout, I will review these crucial concepts / devices, since they will also play a pivotal role in our initial explorations of intensional semantics…**