A Review of the Essentials of Extensional Semantics [1]

1. The Big Picture

(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. A finite number of primitive meaningful units (lexemes, lexical items)

b. A 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

(2) The Principle of Compositionality (Gottlob Frege)

The meaning of a complex expression can be effectively computed from the meaning of its component expressions and their 'mode of combination'.

(3) The Ultimate Goal, Narrowed and Refined

A precise, formal theory of the combinatoric system (algorithm) that our cognitive systems employ to derive/compute the meanings of complex expressions from the meanings of their component parts?

(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?

b. How does this combinatoric system vary across languages? Do languages differ in how they 'compute meanings', and if so, in what ways?

Fact: These questions don't receive nearly as much attention within semantics that they do in syntax and phonology (but this is improving…)

[1] Sections 1-4 are based on material in Heim & Kratzer (1998: 1-12, 13-26), Chierchia & McConnell-Ginet (2000: 1-33, 53-73, 99-104), Larson (1995: 361-368), and Partee (1995: 311-316). Sections 5-8 are based on the bulk of Heim & Kratzer (1998).

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 "The president" and "golfs" such that 'combining them together' gives us the '**meaning**' of "The president golfs"?

The Central Problem:

- The word 'meaning' is a vague, pre-theoretic term from every-day discourse.

  (like 'alive', 'hot' and 'heavy')

- Thus, it may not be an appropriate term for a precise, scientific study of human language.

  (*e.g.* biology doesn't actually employ terms like 'alive' or 'dead' or 'life-form')
  (*e.g.* physics doesn't actually employ terms like 'hot', 'heavy', 'fast', *etc.*)

(6) New, Preliminary Objective

Let's replace our everyday, pre-theoretic concept of 'meaning' with something more precise, more narrow, 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 interested in, 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 (exactly) do we *know* when we *know* 'the meaning' of a sentence?
- Well, we know a lot of things!

a. Social Appropriateness of the Statement:
What social contexts the statement is appropriate in.

"That's wonderful." vs. "That kicks ass!"

b. 'Emotional Content' of the Statement:
What the statement reveals about the emotional state of the speaker.

"I disagree with Dave's judgment." vs. "Dave is a damn fool!"

c. ***The Informational Content of the Statement***
What information about the world the sentence '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. Thus, 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 is not so simple either:

There seem to be different *ways* that information can be 'conveyed' by a sentence.

(8) **Example Dialog**

Person 1: How did your brother's physical go?
Person 2: Well, he's stopped smoking.

Person 2's utterance 'conveys' all the following information:

- His brother has stopped smoking.
- His brother *has been smoking*.
- His brother's physical did not go well. (*i.e.*, bad news at his physical.)

Each of these different bits of information is 'conveyed' in a different way by the utterance.

(9) **Assertion**
The information that *Dave has stopped smoking* is **asserted** by the utterance / speaker

Informal Definition:
Assertion = the information that is 'explicitly added' by the utterance

Linguistic Test:
Sentence S **asserts that** *p* = S is true if and only if *p*

Example: "Dave stopped smoking" is true if and only if Dave stopped smoking.

(10) **Presupposition**
The information that *Dave has been smoking* is **presupposed** by the utterance / speaker

Informal Definition:
Presupposition = the information that is 'taken for granted' by the utterance

Linguistic Test:
Sentence S **presupposes** *p* = S is true *or false* only if *p*
S *and negation of S* is true only if p

Example: "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**
The info that *D's physical didn't go well* is an **implicature** of the utterance / speaker

Informal Definition:
Implicature = the information that is not explicitly asserted in the utterance, but which the speaker (clearly) intends the listener to conclude.

Linguistic Test:
p is **an implicature** of S = *p* is 'conveyed' by S, but 'not *p*' is consistent with S

Example: "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) **Our Goal (Restated)**

A precise, formal theory of the combinatory system that derives the *assertions* of a complex (declarative) sentence from the 'meanings' of its component expressions…

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) **Our Goal (Restated Again)**
A precise, formal theory of the combinatory system that derives the *truth conditions* of a sentence from the 'meanings' of its component parts.

A Quick Review of How We Got Here:

- We want to theorize about how the 'meaning' of a sentence is computed from the 'meaning' of its parts.
- This requires us to make more precise what we mean by 'meaning'.
- This leads us to the notion of *the information that a sentence conveys*
- This requires us to make more precise what we mean by 'conveying information'
- This leads us to the notion of *the information 'asserted' by a sentence*.
- This notion can be recast as *the truth conditions of a sentence*.
- **Thus, we want to know how the *truth conditions* of a sentence can be derived from the 'meanings' of its component parts…**

An Important Reminder:
As we've seen, 'truth-conditions' aren't all there is to the general phenomenon of 'meaning'.
At some point, we will have to come back to the other phenomena in (7), (10) and (11)….

(15) **The 'Psychological Reality' of Truth-Conditions**

Our goal in (14) entails that *part of our cognitive capacity as speakers of a language is a system that derives* ***truth conditions****.*

This isn't so far fetched a claim… consider the following (plausible) characterization of the information computed during a typical conversation…

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

a. Speaker's Utterance: / ðə haʊs ɪz an fajɹ /

b. Listener's Computations:

(i) *Syntax*: The string /ðə haʊs ɪz an fajɹ/ 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…
…oh my god the house is on fire!...

In the model of human communication in (16), a central step in comprehension is the computation of the truth-conditions of the speaker's utterance:

- If listeners have reason to believe that a speaker's utterance is *true* (*e.g.* because the speaker is honest and knowledgeable)….
- …then knowing the **truth conditions** of a sentence allows the listener to thereby deduce *facts about the world!*

CONCLUSION:
To the extent to which (16) is an accurate characterization of the kind of information our systems compute when we 'understand' an utterance, ***then part of what our language systems 'do' during sentence comprehension is compute truth-conditions***

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

a. The Object Language: The language we are *describing* (not using).

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

We want our theory to associate sentences of the object language with their truth conditions, *as characterized in the meta-language*.

- The object language and meta-language can both be *the same language*….

  "The house is on fire" is T (true) *iff* the house is on fire.
  Object language: English *Meta-Language: English*

- …or, they can be two different languages:

  "Wé hít yei kanagán" is T (true) *iff* the house is on fire.
  Object language: Tlingit *Meta-Language: English*

---

3. Obtaining a Theory of Truth Conditions from 'Extensions'

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

(18) **Crude Sketch of a Compositional Semantic Theory that Yields Truth Conditions**

SEMANTICS('The president') + SEMANTICS('golfs') =

SEMANTICS('The president golfs') =

TRUTH-CONDITIONS('The president golfs')

Some Important Questions:

- How do we construct a theory that obtains this result for us?
- What are the 'meanings' of "The president" and "golfs" such that combining them together can yield for us the truth conditions of "The president golfs"?

... more in a moment!

3.1 More about the Meaning of 'Meaning': An Excursus of 'Extensions'

We've already seen that the everyday word 'meaning' is vague and confusing in a number of ways…*here's another:*

(19) **The Meaning of the Phrase 'The President'**

a. In one sense, the *meaning* of the NP "the president (of the USA)" just recently changed. It went from *meaning* Barack Obama to *meaning* Donald Trump. ('denotation', 'reference')

b. In another sense, the *meaning* of the NP is the same now as it was in 2016. It still *means* 'the person who holds the office of the presidency of the United States'. ('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**

a. The *extension* of an NP is the thing in the actual world that the NP refers to.

- The *extension* of "the president" is *Donald Trump.*

b. The *intension* of an NP is the '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 person who holds the office…*

So, the 'meaning' of an NP can be broken up into its 'extension' and its 'intension'…

…can the meaning of a sentence likewise be broken up in this way?

Well, since these are technical terms we're inventing, let's just make something up that works!

(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 'conceptual/propositional 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 (akin to) 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.*

Side-Note:
The (crucial) idea that the extension of a sentence is its *truth value* is one of those 'weird ideas' that you just have to get used to…
See Chierchia & McConnell-Ginet (2000: 62-65) for more discussion and defense…

(23) **The General Picture**

a. Intension of "X":
A kind of 'concept'/ 'formula' / 'function' 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. Illustrative Paradigm:

i. Intension of "the president" = *whoever holds the office of the presidency*

ii. Extension of "the president" = Donald Trump

iii. Intension of "The president golfs" ≈
"The president golfs" is T iff Donald Trump golfs

iv. Extension of "The president golfs" = TRUE

What's the point of all of this?
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]] = the extension of "X"
(function from X to its extension)

(25) **The Compositionality of Extensions**

- We've broken down the 'meaning' of a S/NP into its *extension* and its *intension*.
- 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.
- *Thus, the extension of a complex expression should be derived from the extensions of its component parts!*

  In A Picture: [[The president]] + [[golfs]] = [[The president golfs]] = True

(26) **The Extension of a Predicate = Function**

- In order to make the 'picture' in (25) work, the extension of "golfs" must be such that 'combining' it with the extension of "the president" gives the extension of "The president golfs"…
- *How can we model this?* Well, we know the following:

  (i) [[The president]] = Donald Trump
  (ii) [[The president golfs]] = T(rue)

- *Thus:* [[golfs]] + Donald Trump = T
  The extension of "golfs" 'combines' with Donald Trump to yield T
- *So, the extension of "golfs" is like a 'device' that takes Donald Trump as input and yields T as output…*
- ***So, the extension of "golfs" is like a FUNCTION!***
- The Core Idea:
  [[golfs]] = A function from entities to truth values, which yields T iff that entity golfs.
  = [λx : x golfs]

(27) **Interim Summary: Our Lexical Entries**

a. [[The president]] = Donald Trump
b. [[golfs]] = [λx : x golfs]

We've almost got the picture in (25) worked out… all we need is a rule for 'combining' the extensions of "The president" and "golfs" to yield the extension of "The president golfs."

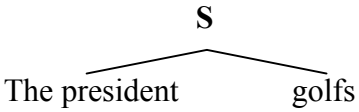
(28) **The Rule of Function Application [FA]**

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 "The president golfs" from (i) the extension of its component pieces, and (ii) the syntax of the sentence.

(29) **Computing the Extension of 'The president golfs'**

a. Simplified Syntactic Assumption:
The structure of the sentence *The president golfs* is as follows:



b. Semantic Derivation:

(i)	[[S]]	=	(by FA)
(ii)	[[golfs]] ([[the president]])	=	(by (27a))
(iii)	[[golfs]] (Donald Trump)	=	(by (27b))
(iv)	[λx : x golfs] (Donald Trump)	=	(by the facts of the world)
(v)	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 *iff* B and B *iff* C **entails** A *iff* C

b. *Substituting 'equals' for 'equals'* If x = y, then x = z *iff* y = z

(31) **Deriving the Truth Conditions of a Sentence via Our Theory of Extensions**

To Prove: "The president golfs" is T iff Donald Trump golfs

a. "The president golfs" is T *iff* (by Syntactic Assumptions)

b. " **S** " is T *iff* (by definition of our notation)

The president golfs

c. [[**S**]] = T *iff* (by FA)

c. [[golfs]]([[the president]]) = T *iff* (by (27))

d. [λx : x golfs] (Donald Trump) = T *iff* (by definition of our λ-notation)

e. Donald Trump golfs.

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!!!**

(32) **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 'golfs', 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 came to this conclusion:

(34) **Determining the 'Meanings' 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, as well as the semantic rules we have.

c. Based on (a) and (b), develop an entry for L which would – in combination with the already established ingredients in (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 'golf' and 'child', however, this approach admittedly ignores (or can ignore) much of their intuitive 'meaning'
  (the entry in (27b) doesn't say anything about what distinguishes 'golfing' from 'bowling'.)

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. Donald Trump)
- Truth Values (e.g. True, False)
- Functions from Entities to Truth-Values (e.g. [λx : x golfs])

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* = 'entity')

b. *t* is a semantic type (*t* = 'truth value')

(37) **The Theory of Semantic Types, Part 2: Functional Types**

If α is a semantic type, and β is a semantic type, then <α,β> is a semantic type.
(<α,β> = 'function from things of type α to things of type β')

(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 [functions from entities to truth-values] to functions from [functions from entities to truth-values] to truth values)

(39) **Notations for Domains**
If α is a semantic type, then D_{α} is the set of all things of type α

- D_{e} = the set of entities
- $D_{<et>}$ = the set of functions from entities to truth-values
- $D_{<et,et>}$ = the set of functions from [functions from entities to truth-values] to [functions from entities to truth values]

6. The Lambda Notation for Functions

In Section 3, we represented [[golfs]] as '[λx : x golfs]'. Let's 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 consequence of how our notation is defined…
Since we'll be using it quite a bit, it's nice to have a name for it: 'Lambda Conversion'

[λx : x ∈ D . φ(x)](a) = φ(a)

(45) **Examples**

a. [λx : x ∈ { 0, 1, 2, 3 } . x + 3](2) = **2 + 3** = 5
b. [λx : x ∈ { Seth, Rajesh } . the office of x](Rajesh) = **the office of Rajesh** = 418

(46) **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!

(47) **Example** [2]

[λx : x ∈ **N** . [λz : z ∈ **N** . x + z]]
This is the function which takes a number x as argument and returns
the function which takes a number z as argument and returns x + z

(48) **Crucial Question / Problem**
- How do we represent a function like [[golfs]] using λ-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 golfs)

(49) **Lambda Notation, Part 2** [3]

a. *Syntax*: [λx : x ∈ D . **IF** φ(x) **THEN** y, **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 ∈ D,
maps x to y *if* φ(x),
and maps x to z otherwise

(50) **Example**

[λx : x ∈ D_e . **IF** x golfs **THEN** T, **ELSE** F] =

a. The function whose domain is D_e, and for all x ∈ D_e, maps x to T *iff* x golfs
b. [[golfs]]

[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

(51) **Another Example**

[λy : y ∈ D_e . [λx : x ∈ D_e . **IF** x likes y **THEN** T, **ELSE** F]] =

a. The function whose domain is D_e and for all y ∈ D_e, maps y to…
the function whose domain is D_e, and for all x ∈ D_e, maps x to T *iff* x likes y

b. [[likes]]

(52) **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:

[λy : y ∈ D_X . ….] = (i) [λy ∈ D_X : ….]
(ii) [λy_X : ….]
(iii) *when it's clear from context what the domain of the function is, we can even just write:* [λy : …]

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

- Note that, given our definition in (40), if 'φ(x)' is a meta-language *sentence*, then it *isn't* immediately clear what '[λx : x ∈ D . φ(x)]' should mean.

Example: [λx : x ∈ D_e . x golfs] =
'the function that maps an entity x onto… *x golfs?!?*
…what the heck is 'x golfs'?

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

The Abbreviatory Convention:
If 'φ(x)' is a meta-language *sentence* then '[λx : x ∈ D . φ(x)]' is short-hand for
'[λx : x ∈ D . **IF** φ(x) **THEN** T, **ELSE** F]

Example: [λx : x ∈ D_e . x golfs] =
[λx : x ∈ D_e . **IF** x golfs **THEN** T, **ELSE** F] =
'The function that maps an entity x to T *iff* x golfs'

NOTE:
- The abbreviatory convention in (53) is not part of standard λ-notation.
- However, since its introduction in Heim & Kratzer (1998), it has become rather wide-spread throughout natural language semantics.

7. Indices and Assignment Functions

(54) **Question**

- How should / could our system model the meanings of *pronouns* (e.g., "he")*?*
- What (if anything) should our semantic valuation function "[[.]]" yield as the extension of a pronoun (e.g., "he")?

(55) **Our Procedure for Framing Hypotheses about Lexical Items (Section 4)**

a. Consider the truth-conditions of sentences in which "he" 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 "he" which would – in combination with the entries for the other words in the sentences (b) – correctly derive the truth conditions of the sentences "he" appears in (a).

(56) **Core Data / Observations**

a. Fact 1:
The extension of a pronoun seems to be an entity.

- *In a context where the speaker is pointing to Barack:*
  [[He golfs]] = T *iff* Barack golfs.

- Given that [[golfs]] = [λx : x golfs], this suggests [[he]] = Barack

(b) Fact 2:
The extension of a pronoun isn't a *fixed* entity, even in a single context.

- *In a context where the speaker points first to Barack, and then to Joe.*
  [[He golfs and he dances]] = T iff Barack golfs and Joe dances.

- Thus, given our 'procedure' in (55), this suggests that (paradoxically) [[he]] = Barack and [[he]] = Joe

(57) **The Challenge**

- Since "[[.]]" is by assumption a function, it cannot yield two different values for the same input.

- Thus, it can't simultaneously be that [[he]] = Barack and [[he]] = Joe.

The standard solution/analysis comes in two steps, a syntactic one and a semantic one.

(58) **The Syntactic Step: Indices**

Pronouns must bear an *index*.
- We can take an index to be a natural number.
- We can represent the pronoun 'bearing' the index via sub-scripting

Illustration: he_1 , him_2 , she_{34} , it_{105} , *her

(59) **The Semantic Step: Interpretation of Indices**

a. Assignment Functions
- Indices are interpreted. However, indices are not interpreted by "[[.]]" but by a special function (usually represented "g"), called an *assignment function.*
- An *assignment function* is any function from indices to entities.

Examples: g = { < 1 , Dave > , < 2 , Barack > , < 3 , Joe > … }
h = { < 1 , Lou > , < 2 , Mitt > , < 3 , Paul > … }

b. Pairing "[[.]]" Within an Assignment Function
- Our semantic valuation function "[[.]]" now always comes paired with some assignment function.
- If we are pairing "**[[ ]]**" with the assignment function *g*, we write " $\mathbf{[[\ .\ ]]^g}$ "

(60) **The Pronouns (and Traces) Rule [PR]**
Let g be any assignment function. If X_i is a pronoun (or a trace) bearing index *i*, then $[[X]]^g = g(i)$.

(61) **Illustration of the Solution**

Let *g* be the assignment function defined in (59a). Thus…

a. $[[\ [\ he_2$ golfs] and [he_3 dances] $]]^g$ = T *iff* (by FA, Lex., LC)

b. $[[golfs]]^g([[he_2]]^g)$ = T and $[[dances]]^g([[he_3]]^g)$ = T *iff* (by PR)

c. $[[golfs]]^g(g(2))$ = T and $[[dances]]^g(g(3))$ = T *iff* (by def. of 'g')

d. $[[golfs]]^g$(Barack) = T and $[[dances]]^g$(Joe) = T *iff* (by Lex.)

e. [λx : x golfs](Barack) = T and [λx : x dance](Joe) = T *iff* (by LC)

f. Barack golfs and Joe dances.

8. Summary of the Semantic Composition Rules from LING 610

We now have enough on the table to summarize all the semantic composition rules that you learned last semester.

- It wouldn't hurt to review your notes and materials from last semester, to refresh your memory on how these rules can be used to derive the extensions of the following kinds of constructions:
  - Transitive verbs (and ditransitive verbs)
  - Adjectival modification
  - Definite descriptions (assuming the 'partial function' semantics for *the*)

- The most complex of these rules is perhaps Predicate Abstraction [PA] in (66). Definitely spend some time reviewing your notes on how this rule allows our system to capture the semantics of:
  - Movement structures and relative clauses
  - Pronominal binding
  - Quantifiers in non-subject position

(62) **Function Application [FA]**
Let g be any assignment function. If X is a branching node that contains two daughters – Y and Z – and if $[[Y]]^g$ is a function whose domain contains $[[Z]]^g$, then:

$$[[X]]^g = [[Y]]^g ([[Z]]^g)$$

(63) **Predicate Modification [PM]**
Let g be any assignment function. If X is a branching node that contains two daughters – Y and Z – and if both $[[Y]]^g$ and $[[Z]]^g$ are in $D_{<et>}$, then:

$$[[X]]^g = [\lambda x : x \in D_e . [[Y]]^g(x) \text{ and } [[Z]]^g(x)]$$

(65) **The Pronouns (and Traces) Rule [PR]**
Let g be any assignment function. If X_i is a pronoun (or a trace) bearing index *i*, then:

$$[[X]]^g = g(i).$$

(66) **Predicate Abstraction [PA]**
Let g be any assignment function. If X is a branching node whose daughters are Y and Z, and if Y is the index *n*, then:

$$[[X]]^g = [\lambda x_e : [[Z]]^{g(n/x)}]$$

Note:
For those who are interested, I've also put together a quick review of PA and how it can be used to provide a semantics for movement constructions and pronominal binding…