LING 510, Lab 5
October 7, 2013

Agenda:

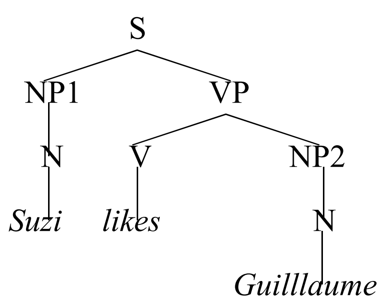
- How to finish computing the truth conditions of a sentence.
- Computing the truth conditions of sentences with conjunction.

Key point from last week: Lambda notation

- Lambda notation offers a very handy and simple way of defining functions that *yield other functions as values*. Our old notation was clunky and required a lot of space on the page.
- The way to represent such functions is incredibly simple: *You just embed one lambda formula inside another one*.

1. How to finish the derivation of a sentence's truth conditions

(1) Compute the truth conditions of the sentence *Suzi likes Guillaume.*



Note that I **haven't** asked you to compute the **truth value** of this sentence. You don't know if this sentence is true, so you couldn't compute its truth value, even if I had asked you to. You **can**, however, compute its truth conditions (what the world would have to be like in order for this sentence to have a truth value of 1).

Lexical entries:

i. [[*Suzi*]] = Suzi
ii. [[*likes*]] = $\lambda x: x \in D_e. [\lambda y: y \in D_e.$ y likes x]
iii. [[*Guillaume*]] = Guillaume

Subproofs:

iv. Subproof of NP1
a. [[NP1]] = [[N]] = [[*Suzi*]] — by NN x2
b. [[*Suzi*]] = Suzi — by TN and i

v. Subproof of NP2

a. [[NP2]] = [[N]] = [[*Guillaume*]] — by NN x2

b. [[*Guillaume*]] = Guillaume — by TN and iii

vi. Subproof of V

a. [[V]] = [[*likes*]] — by NN

b. [[*likes*]] = $\lambda x: x \in D_e.\ [\lambda y: y \in D_e.$ y likes x] — by TN and ii

vii. Subproof of VP

a. [[VP]] = [[V]]([[NP2]]) — by FA

b. [[V]]([[NP2]]) =
$\lambda x: x \in D_e.\ [\lambda y: y \in D_e.$ y likes x](Guillaume) — by v and vi

c. $\lambda x: x \in D_e.\ [\lambda y: y \in D_e.$ y likes x](Guillaume) =
$\lambda y: y \in D_e.$ y likes Guillaume — by LC

Truth conditions of S:

a. [[S]] = [[VP]]([[NP1]]) — by FA

b. [[VP]]([[NP1]]) =
$\lambda y: y \in D_e.$ y likes Guillaume (Suzi) — by iv and vii

How to know when it's time to determine the truth conditions (the last step):

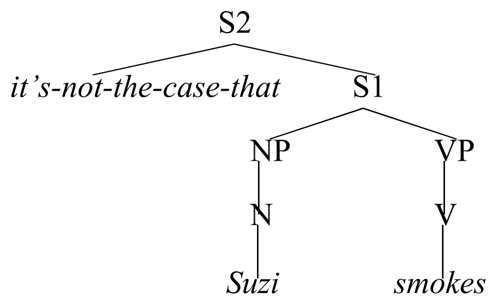
- If you are working on the biggest S node…
- …and you find that there is just one more argument to be taken by your function…
- …then go to the next line and rewrite what you wrote after the last = sign. You will be figuring out the truth conditions for the expression that you just wrote.
- **Truth conditions for a sentence S are written as follows:**
  (…denotation of S…) = 1 *iff* THE WORLD IS LIKE THIS.
- **Pro-tip:** The part that I've written in SMALL CAPS can generally be replaced by the sentence you would produce if you did the last step of Lambda Conversion and put Suzi in for y in the formula above.

In this example, the expression whose truth conditions you're figuring out is the following:
$\lambda y: y \in D_e.$ y likes Guillaume (Suzi)

We know that [[S]] = $\lambda y: y \in D_e.$ y likes Guillaume (Suzi). We know what the world would have to be like in order for this sentence to be true (Suzi would have to be like Guillaume). Therefore, we can write the truth conditions for S as below. This is the last step of the derivation. **Thisf last step is always attributed to Lambda Conversion.**

c. $\lambda y: y \in D_e.$ y likes Guillaume (Suzi) = 1 *iff* Suzi likes Guillaume — by LC

(2) Compute the truth conditions of the sentence *It is not the case that Suzi smokes.*



<u>Lexical entries:</u>

i. [[*Suzi*]] = Suzi

ii. [[*smokes*]] = λy: y $\in$ D_e.y smokes

iii. [[*it is not the case that*]] = λp: p $\in$ D_t.p = 0

Let's say that you got as far as the following steps in your derivation:

iv. <u>Subproof for S1:</u>

→ **Every time you hit a S node, derive its truth conditions.**

a. [[S1]] = [[VP]]([[NP]])	by FA
b. [[VP]]([[NP]]) = λy: y $\in$ D_e.y smokes (Suzi)	by TN, i, and ii
c. λy: y $\in$ D_e.y smokes (Suzi) = 1 *iff* Suzi smokes	by LC

<u>Truth conditions of S2:</u>

a. [[S2]] = [[*it's not the case that*]]([[S1]])	by FA
b. [[*it's not the case that*]]([[S1]]) = λp: p $\in$ D_t.p = 0 ([[S1]])	by TN, iii

It is time to determine the truth conditions!

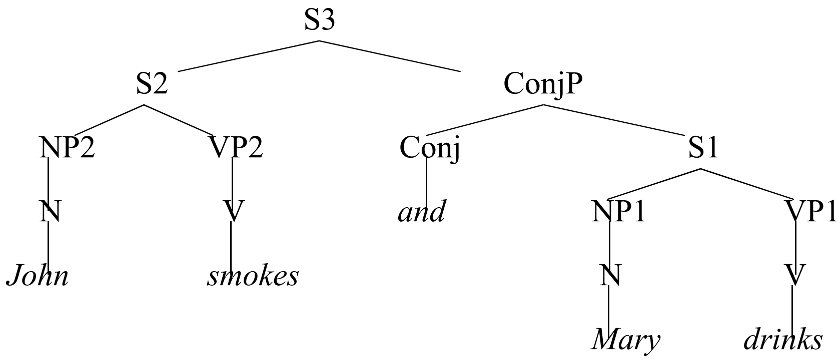
c.	by _______

Note!

When one of the arguments of a function f (like negation) is the extension of a sentence (AKA, a truth value), e.g., S1, you can either write f([[S1]]) or f([[*Suzi smokes*]]) or f(λy: y $\in$ D_e.y smokes (Suzi)). Ilaria and I usually write it the first way because it is shortest.

Don't write f(Suzi smokes). A function like negation doesn't take an English string as its argument, it takes a truth value.

2. Sentential conjunction



1. What is the type of sentential conjunction?

2. What is the denotation of sentential conjunction?

[[*and*$_S$]] =

3. Compute the truth conditions of *John smokes and Mary drinks*.

<u>Lexical entries:</u>

i. [[*John*]] = John ii. [[*Mary*]] = Mary
iii. [[*smokes*]] = λx. x smokes iv. [[*drinks*]] = λy. y drinks
v. [[*and*$_S$]] = λp: p ∈ D_t. [λq : q ∈ D_t .p = 1 and q = 1]

<u>Subproofs:</u>

<u>vi. Subproof for S1</u>

→ **Every time you hit a S node, derive its truth conditions.**

a. [[S1]] = [[VP1]]([[NP1]]) by FA
b. by _____

c. by _____

<u>vii. Subproof for S2</u>

a. [[S2]] = [[VP2]]([[NP2]]) by FA
b. by _____

c. by _____

viii. Subproof for ConjP

a. [[ConjP]] =

b.

c.

Truth conditions of S3:

a. [[S3]] =

b.

c.

3. Summary of Deriving Truth Conditions

- How to know when it's time to determine the truth conditions (the last step)…and what to do:
  - If you are working on the biggest S node…
  - …and you find that there is just one more argument to be taken by your function…
  - …then go to the next line and rewrite what you wrote after the last = sign. You will be figuring out the truth conditions for the expression that you just wrote.

- Truth conditions for a sentence S are written as follows:
  (…denotation of S…) = 1 *iff* THE WORLD IS LIKE THIS.

- The last step of the derivation where you derive the truth conditions for the biggest sentence is always attributed to Lambda Conversion (LC).

- Every time you hit a S node, derive its truth conditions. We want to see this in your subproofs for S nodes.

- When one of the arguments of a function f is a truth value, e.g., [[S2]] (which we'll say is the sentence *John smokes*), you can write f([[S2]]), f([[*John smokes*]]), or f(λx. x smokes(John)).
  - Don't write f(John smokes). **Why not?**