Ling 130 Notes: What can lambdas do for you

Sophia A. Malamud
March 15, 2011

1. Phrasal conjunction

John walks and Mary talks
Every boy and some girls laugh <=> Every boy laughs and some girls laugh
John walks and talks <=> John walks and John talks
Some man walks and talks <> Some man walks and some man talks

The conjunction schema:
For M, N expressions of type <i i>
[[ M and N ]] = \lambda X. M(X) & N(X)

2. Syntactic movement, traces, and lambdas

Syntax has three parts: lexicon; phrase structure rules, and movement rules.

Part 1. Lexicon:

N \rightarrow man, woman, apple, boy, girls
Npr \rightarrow John, Mary
V \rightarrow walks, talks, laugh(s), loved, run(s), kiss(es), sink(s)
Det \rightarrow every, some, the, a

Part 2. Phrase structure rules:

S \rightarrow NP VP
NP \rightarrow Det N'
N' \rightarrow Adj N'
N' \rightarrow N' PP
...
Part 3. Movement rules – used to be known as “transformations”
They apply when we find an element displaced somewhere where we don’t expect it.

Example 1: topicalization
John, I like. from I like John.
Usually, the object follows the verb!

Example 2: question formation
Is John here from John is here
Who do you want to invite from something like You want to invite who
  • We don’t want to permit things to move any which way – we would lose all
generalizations on how words are ordered.
    ◦ One constraint: things only move when they absolutely have to, to satisfy some
ingeneral principle.
    ◦ Another constraint: all movement leaves traces – invisible “ghosts” of things
moved. (we co-index the moved thing and its trace, to show what’s what).

Further constraints can be (and are) stated as restrictions on the occurrence of traces and co-
indexed moved things.

So, Who do you want to invite is actually Who_i do you want to invite t_i

How do we know traces are real? They interfere with phonological contraction:
ok Who do you wanna invite  Who do you want to invite t
* Who do you wanna come  Who do you want t to come

Just like actual NPs:
* I wanna John come  I want John to come

How does movement affect semantics?

1. traces are variables: NP trace is always of type e
2. traces are bound by lambdas: a co-indexed lambda is waiting to catch the trace variable higher in the tree (below the moved thing)

So, movement results in lambda-abstraction over the trace.

Example: Some boys, I hate. John, I like.

3. Lambda abstraction and relative clauses

Every man who(m) Mary loved runs.

A bit more syntax: to allow every to apply to man who Mary loved (not just man), we have N’ → N’ Rel

But what is a relative clause? It looks suspiciously like a question: who Mary loved.

So, it’s built from something like a sentence, by moving who to the front:

man who(i) Mary loved t(i)

Interpreting up the tree, we get [[ who Mary loved t ]] = λx. loved(m,x)

set of people such that Mary loved them

Now, relative clauses combine with N’ using a kind of predicate conjunction (like the phrasal conjunction above): [[ N’ Rel ]] = λx. ([[ N’ ]] (x) & [[ Rel ]] (x))

So, [[ man who Mary loved t ]] = λx. man(x) & loved(m,x)

Set of people such that they are men and Mary loved them (note: it’s intersection of two sets)