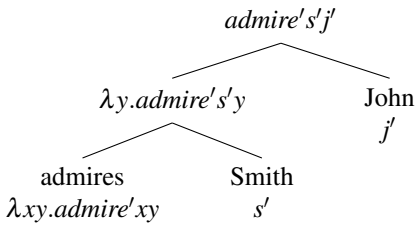
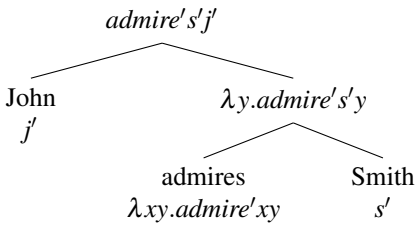
A common way to represent the compositional construction of the meaning of a complex expression is to draw a derivation tree. One type of derivation tree directly represents the applicative structure, obeying the left-right order of function application.

(1) ‖John admires Smith‖ =



Another option is to represent the linear order of the items of the expression as in,

(2) ‖John admires Smith‖ =

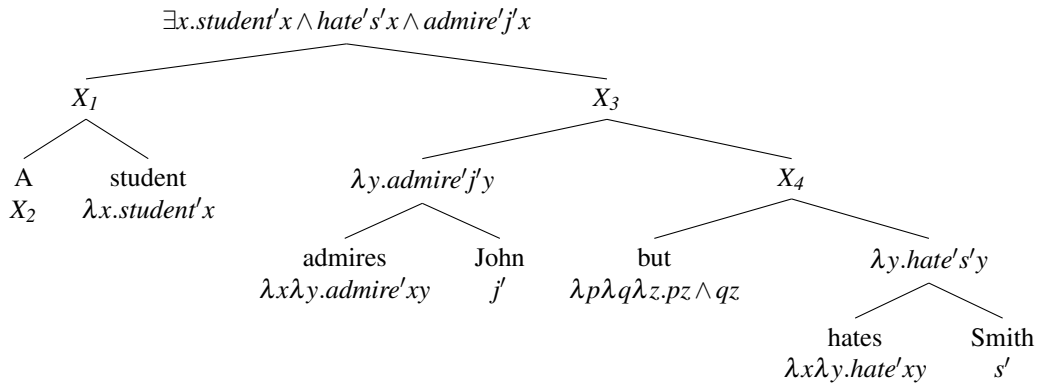


In this option the function-argument order is not represented directly but can be inferred from the types of the expressions merged at a given node. In the rest of this document, we will take the second option.

Q1.

Here is an example we discussed in class:[1]

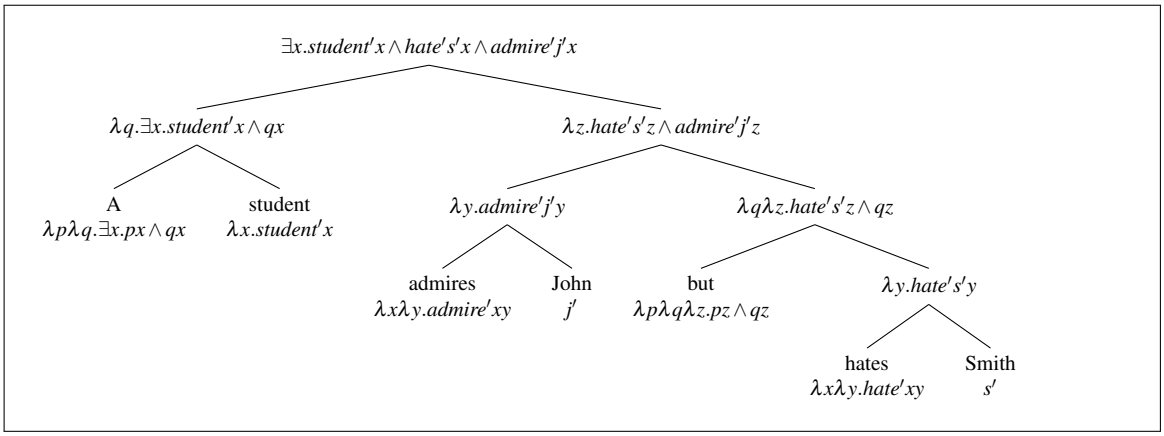
(3) ‖A student admires John but hates Smith‖ =



Specify the missing interpretations X_n.

Solution:

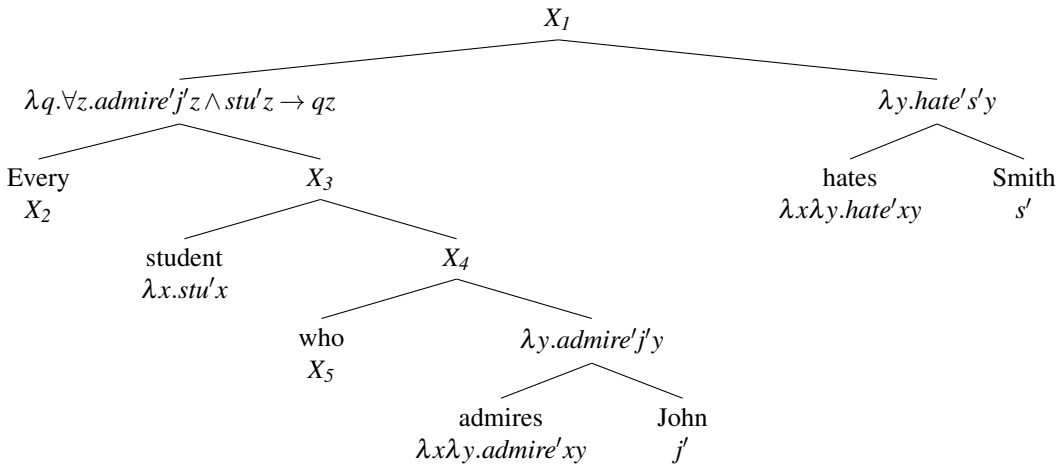
[1] We will take *but* to be logically equivalent to *and*. You should be able to verify that we do not have any means to do justice to the meaning difference between *but* and *and* in the present state of our model theory (sets, membership, and so on). You are invited to think on how to capture the distinction.



Q2.

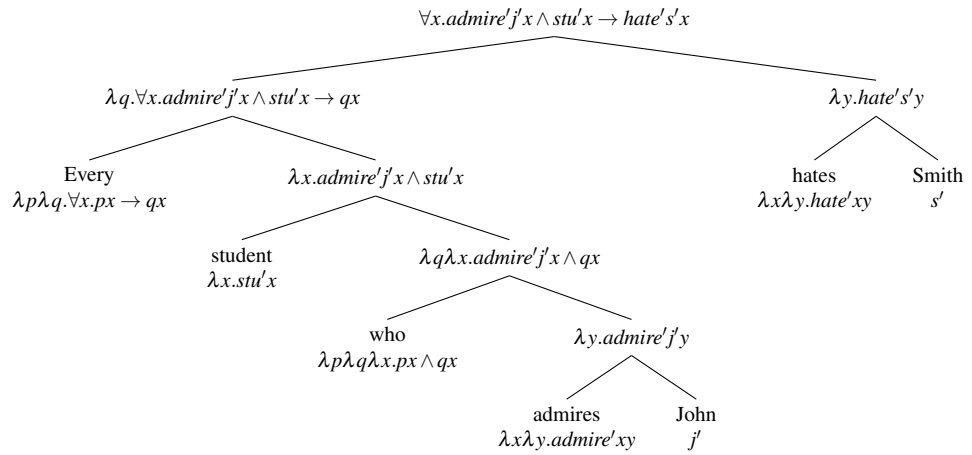
A similar example:

(4) ‖Every student who admires John hates Smith‖ =



Specify the missing interpretations X_n.

Solution:



Q3.

So far we have been taking the world to contain "ordinary" individuals like John, chairs, students, and so

on. A critical observation made by Donald Davidson in his "Logic of Action Sentences", dated 1967, forced semanticists to admit events as a special type of individual in the model. Take the following,

(5) John killed Smith with a knife.

Let's assume we do not need event individuals. Then, the only way left to capture the meaning of (5) is to take *kill* as a three-place relation relating a murderer, a victim and an instrument. The interpretation of the sentence would be something like:

(6) $\|\text{John killed Smith with a knife}\| = \exists x.knife'x \wedge kill'smith'xjohn'$

A serious problem with this interpretation is the impossibility of logically representing the following entailment:

(7) John killed Smith with a knife $\rightarrow$ John killed Smith.

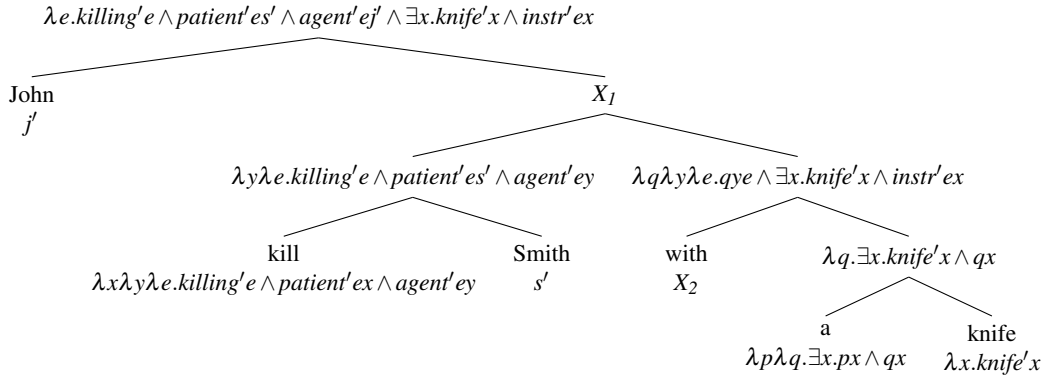
The problem is not limited to instruments; take *John killed Smith with a knife, at his hotel room, quarter past midnight.*

The solution adopted by many semanticists is to interpret sentences like (5) as declarations of the existence of an event, possibly with certain additional properties:

(8) $\|\text{John killed Smith with a knife}\| = \lambda e.killing'e \wedge agent'ej' \wedge patient'es' \wedge \exists x.knife'x \wedge instr'ex$

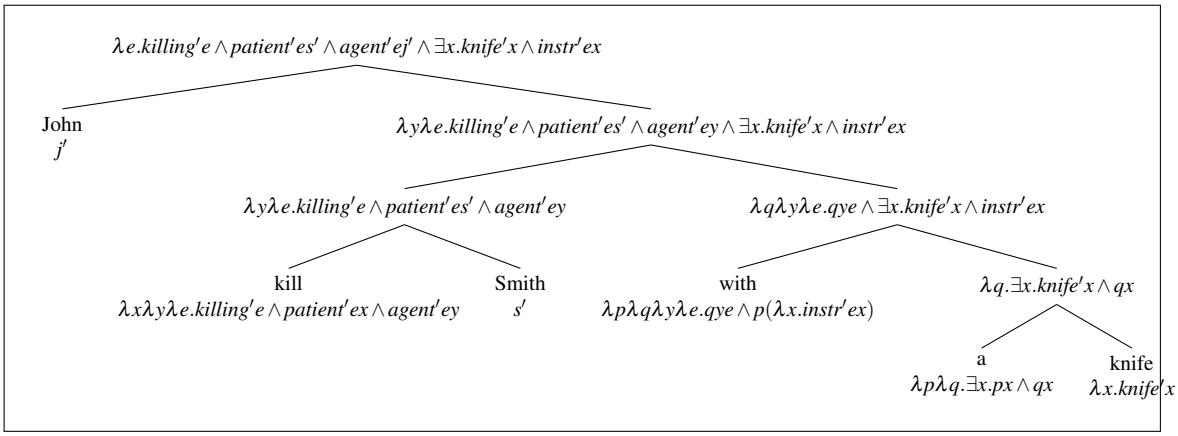
This is a couple of notches simpler than the standard interpretation. For instance, the interpretation denotes a set of events[2] rather than asserting the existence of an event instance. In this respect it is not a satisfactory interpretation of (5), which *asserts* the existence of a particular event. You are invited to think about how to have a fully satisfactory interpretation, but for the present exercise we will ignore this aspect of the sentence, and take it as if denoting a set of events. Any event in the world that would count as John killing Smith by using a knife will be the element of this set.

Here is a derivation tree for the interpretation above. Your task is to fill in the missing slots X_n.



Solution:

[2]Think over and internalize this fact: any lambda term whose fully saturated form denotes a proposition represents a set.


λe.killing′e ∧ patient′es′ ∧ agent′ej′ ∧ ∃x.knife′x ∧ instr′ex
John
j′
λyλe.killing′e ∧ patient′es′ ∧ agent′ey ∧ ∃x.knife′x ∧ instr′ex
λyλe.killing′e ∧ patient′es′ ∧ agent′ey
λqλyλe.qye ∧ ∃x.knife′x ∧ instr′ex
kill
λxλyλe.killing′e ∧ patient′ex ∧ agent′ey
Smith
s′
with
λpλqλyλe.qye ∧ p(λx.instr′ex)
λq.∃x.knife′x ∧ qx
a
λpλq.∃x.px ∧ qx
knife
λx.knife′x