564 Lecture 21 Nov. 4, 1999

1 Homework notes:

1. What we're doing when we calculate the value of a tree like the one "Dino is the brown dinosaur that licked Fred", is we're calculating its *truth-conditions*, i.e. we're finding out what state of affairs must hold in the real world to make that sentence true. So saying that [[**Dino is the brown dinosaur that licked Fred**]] means that Dino is the brown dinosaur that licked Fred, isn't exactly right. Rather, [[**Dino is the brown dinosaur that licked Fred**]] = (means) 1 iff Dino is the brown dinosaur that licked Fred. Here's the difference: Saying that "[[**Dino** ..]] means that Dino is the brown dinosaur that licked Fred" implies that speakers only utter true sentences (or else that they *cause* situations to come about by speaking!). We know that's not the case. Rather than say "[[**Dino** ..]] means that Dino is the brown dinosaur that licked Fred iff the speaker is telling the truth", we say rather that it "[[**Dino** ..]] means true iff Dino is the... ".

2 Review: How To Apply Rules of Interpretation to Branching Nodes

Many people seem to have a problem figuring out whether to apply F.A. or P.M. (or, for that matter, P.A.) when they come to a branching node. Here's some rules of thumb:

(1) *Guess (educatedly) what the types of the daughter nodes are going to be*

We've found, generally, that there's a significant correspondence between syntactic category and type. So, in general, so far:

DP <e>.
VP <e,t>.
NP <e,t>
AdjP <e,t>
CP <e,t>. (because of PA)
V <e,t> (if intransitive)
<e,<e,t>> (if transitive)
<e,<e,<e,t>>>.
[[**the**]] <<e,t>,<e>>
IP <t> (same as "S")

(2) Look at the three rules we have to determine the interpretation of branching nodes, and decide which one applies, *based on the types of the daughter nodes,* β, γ.
If it makes it clearer, use the following metric:

(a) If the daughter nodes are both of type <e,t>, Predicate Modification applies.

(b) If the daughter nodes are something with an index (e.g. a relative pronoun wh_i or $such_i$, or the way some of you did last week's homework, $that_i$) and something of type <t> (e.g. C', IP), then P.A. applies

(c) Otherwise, F.A. applies (and one thing had better be a function with the type of the other thing in its domain, i.e. the part before the comma in the type of the first one had better be the type of the second one. So, in theory, it's certainly possible to have an adjective take a noun as its argument, but in that case, your lexical entry for the adjective had better be of the right type (<<e,t>,<e,t>>)).

(3) **Rewrite the node's denotation** ([[XP]]) **in the template provided by the applied rule**. So, for any node [[XP]], whose daughters are [[YP]] and [[ZP]]:

if F.A. applies:
[[XP]] = [[YP]]([[ZP]])
(if YP is the function taking ZP as its argument, otherwise the other way around)

if P.M. applies:
$[[XP]] = [\lambda x \quad D_e \textbf{ .} [[XP]](x) = [[YP]](x) = 1]$

if P.A. applies
$[[XP]] = [\lambda x \quad D_e \textbf{ .} [[ZP]]^{x/i}]$
(if YP is the indexed relative pronoun, otherwise the other way around)
Important: the letter abstracted by lambda and the letter assigned to the index have to be the same...

(The actual forms of the rules in question:
F. A. *Functional Application*

If α is a branching node and $\{\beta,\gamma\}$ the set of its daughters, then for any assignment a, if $[[\beta]]^a$ is a function whose domain contains $[[\gamma]]^a$, then $[[\alpha]]^a = [[\beta]]^a([[\gamma]]^a)$.

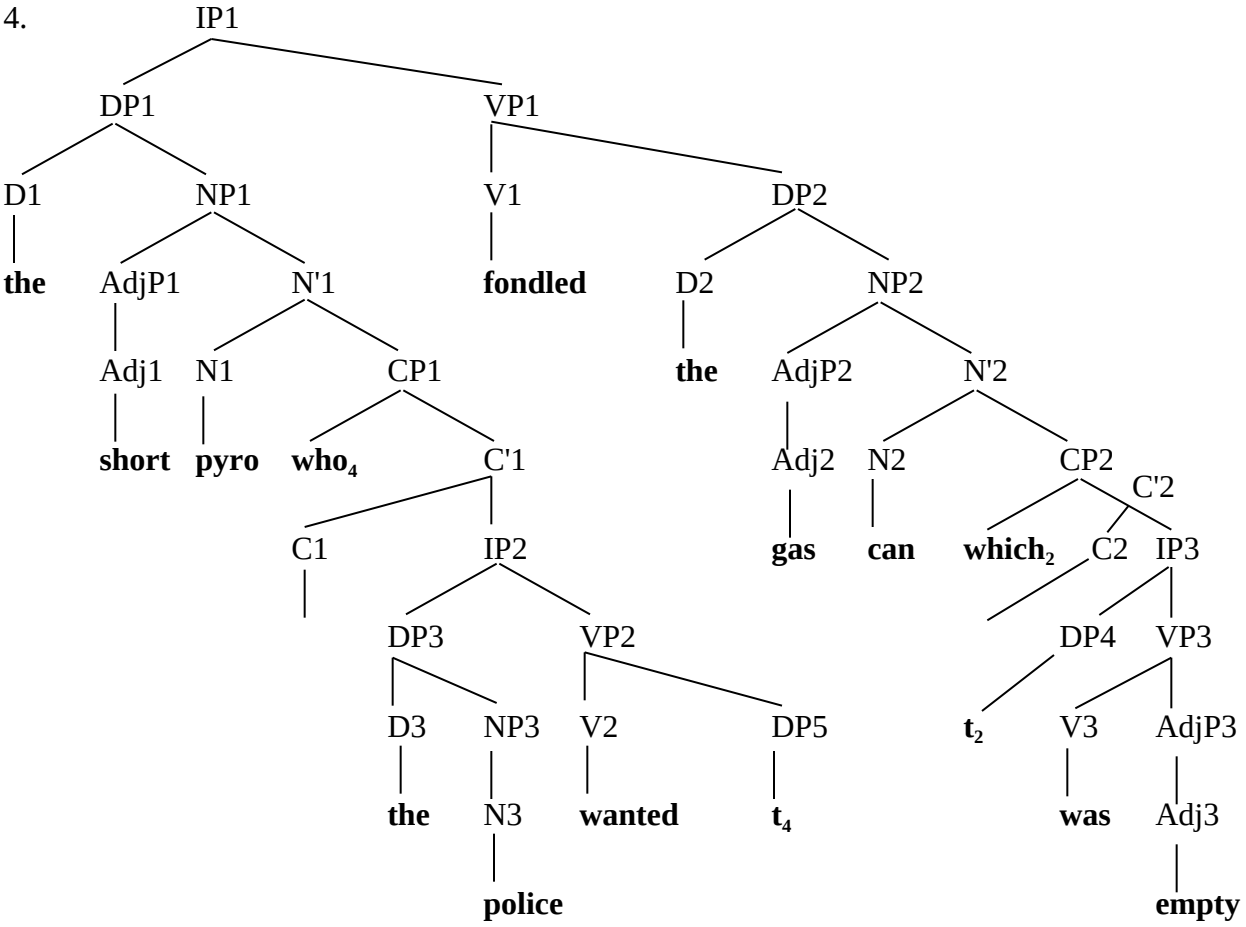
P. M. *Predicate Modification*
If α is a branching node and $\{\beta,\gamma\}$ the set of its daughters, then, for any assignment a, if $[[\beta]]^a$ and $[[\gamma]]^\alpha$ are both functions of type <e,t>, then
$[[\alpha]]^a = [\lambda x \quad D_e \textbf{ .} .[[\beta]]^a(x) = .[[\gamma]]^a(x) = 1]$

P.A. *Predicate Abstraction*

If α is a branching node whose daughters are β_i and γ, where β is a relative pronoun or "such", and i |N, then for any variable assignment a, $[[\alpha]] = [\lambda x \quad D_e . [[\gamma]]^{a^{x/i}}]$

So, class project:
(a) assign types to every node in this tree
(b) for every node, tell me what rule you have to apply to find its denotation (including the non-branching nodes)

4.



Node	Types of daughters	Rule	Node	Types of daughters	Rule
IP1			**the**		
DP1			V2		
D1			**wanted**		
the			DP5		
NP1			$\mathbf{t_4}$		
AdjP1			VP1		
Adj1			NP2		
short			AdjP2		
N'1			Adj2		
N1			**gas**		
pyro			N'2		
CP1			N2		
$\mathbf{who_4}$			**can**		
C'1			CP2		
C1			$\mathbf{which_2}$		
IP2			C'2		
DP3			C2		
D3			IP3		
the			DP4		
NP3			$\mathbf{t_2}$		
N3			VP3		
police			V3		
VP2			**was**		
V1			AdjP3		
fondled			Adj3		
DP2			**empty**		
D2					

3 Review 2: Applying the definition of lambda notation

When you *do* have functional application going on, you're eventually going to have to use the definition of lambda-notation to figure out what the denotation of the node is. Everybody seems to be fine with simply plugging in an individual in the position of the lambda-term, but it can get confusing when there's a lot of stuff happening at once, or when the argument is not an individual but a function.

Let's do some examples:

5.

(a) [λx D_e **.** x is a cat](Felix) = _____

(b) [λx D_e **.** x is calico]([[**Felix**]]) = _____

(c) [λx D_e **.** [λy D_e **.** y chased x]](Rover)(Felix) = _____

(d) [λx D_e **.** [λy D_e **.** y chased x](Rover)](Felix) = _____

(e) [λx D_e **.** [[**calico**]](x)=[[**cat**]](x)=1] = _____

(f) [λf $D_{<e,t>}$ and there is only one x D_e s.t. f(x) =1 **.** the unique y D_e s.t. f(y)=1]([λx D_e **.** [[**calico**]](x)=[[**cat**]](x)=1]) = _____
_____

(g) [λx D_e **.** [λw D_e **.** [λy D_e **.** y is a cat](w)=[λz D_e **.** z chased Rover](w)=1](x)=[λv D_e **.** v is calico](x)=1](Felix) = _____
_____
_____
_____

Homework:

1. For the sentence "Felix is a calico cat who chased Rover", draw the tree and calculate its truth conditions as far as you can without using the Lexical Terminals rule -- i.e., don't fill in the definitions (be they functions or individuals) of the lexical items. (Note that this "is" is not the same as the "is" in "Felix is *the* calico cat who chased Rover"; rather, this is the vacuous *is* we've assumed so far for sentence slike "Felix is a cat" and "Felix is calico".)

2. Fill in the lexical items one by one, applying the definition of the lambda-notation whenever you can as you go -- that is, reduce the terms as soon as you can. (Some terms you'll have been able to reduce somewhat even before you fill in the lexical items: specifically, those created by P.M. and the definition of "the" -- you should have done that in the previous question).

2. Bring a question to class on Tuesday -- and not a question of the "What have you got in your pocketses?" class either!