



(4) **Key Observation 2**

The two ‘special’ components of a relative clause *seem to depend on one another*.

- a. ‘Gaps’ cannot just freely appear in English sentences  
\* Michelle likes.
- b. No relative operator in an English sentence without a gap  
\* a man [ **who** Michelle likes **him** ].

(5) **Extremely Consequential Syntactic Question**

How do we represent the inter-dependence of the relative operator and the gap?

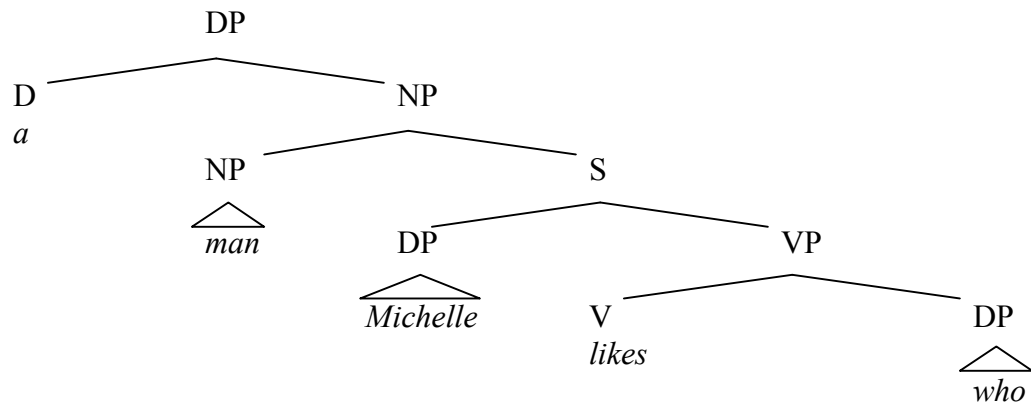
(The answer to this question greatly affects the overall grammatical architecture...)

(6) **The Answer We Will Assume: Movement**

Gaps are created by the ‘movement’ of special expressions, like relative operators.

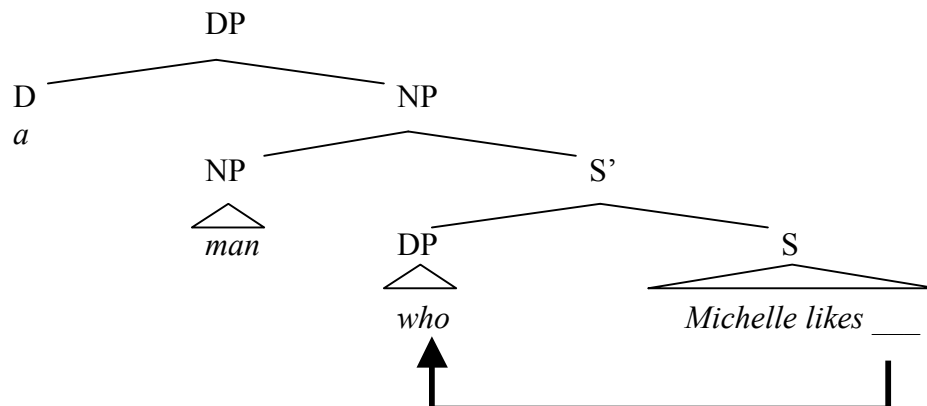
a. The Idea, Part 1

In the syntactic derivation of the sentence, the relative pronoun ‘starts off’ at the position of the gap.



b. The Idea, Part 2

Later in the syntactic derivation, the relative pronoun undergoes *movement*, and is *moved* to a position at the periphery of the relative clause.



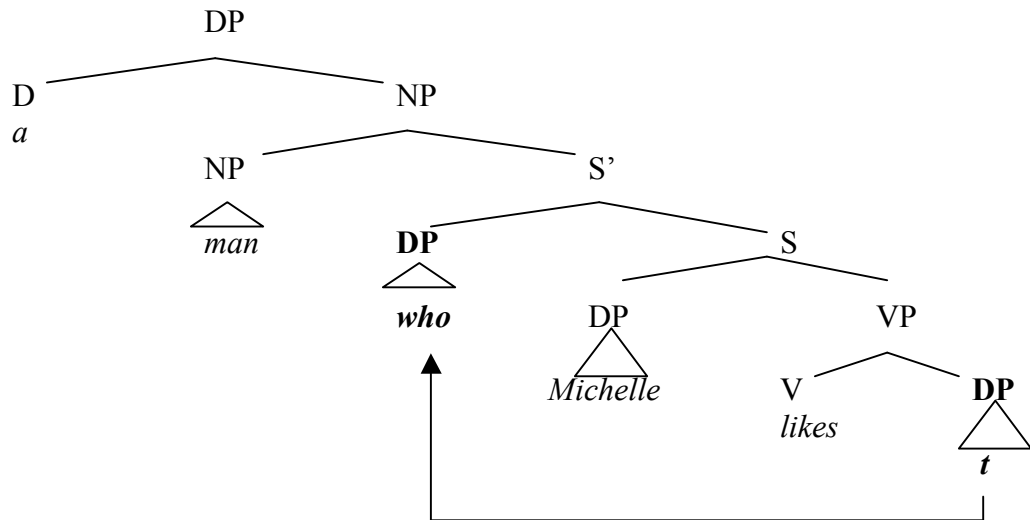
(7) **Some Crucial Further Refinements of the Basic Idea**

The syntactic proposals in (6) have been around in some form or another since at least the 1950s. However, since the 1970's, there have been two key refinements.

a. Traces

When a phrase (*i.e.* the relative operator) is moved, it 'leaves something behind' in the original position: a *trace*.

- When a DP undergoes movement, the position the DP originally occupied is now occupied at all later stages of the derivation by a trace (*t*)
- While a trace *t* is present in the syntax, it is phonologically empty.

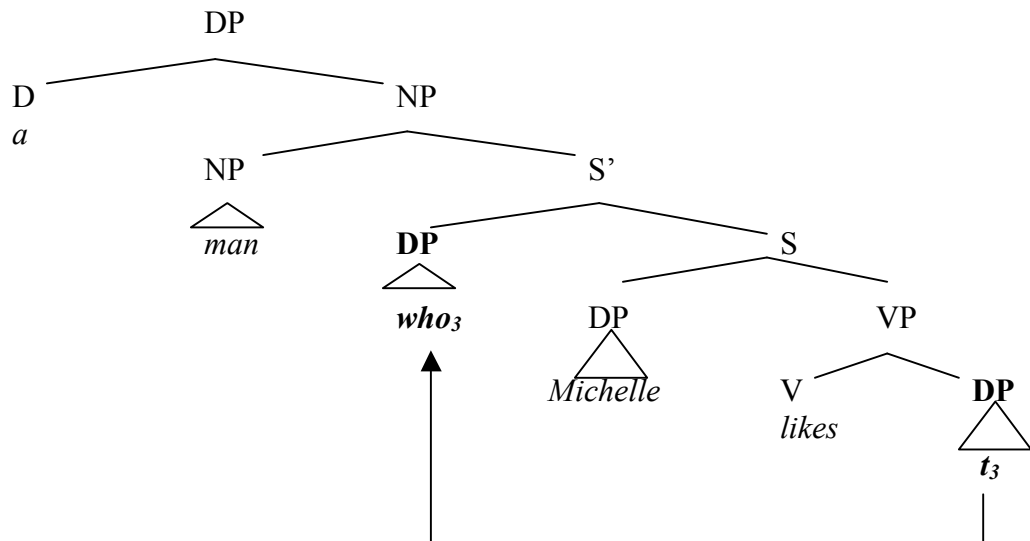


b. Indexing

Like a pronoun, the relative operator bears an index.

Like a pronoun, the trace left by movement of the relative operator bears an index.

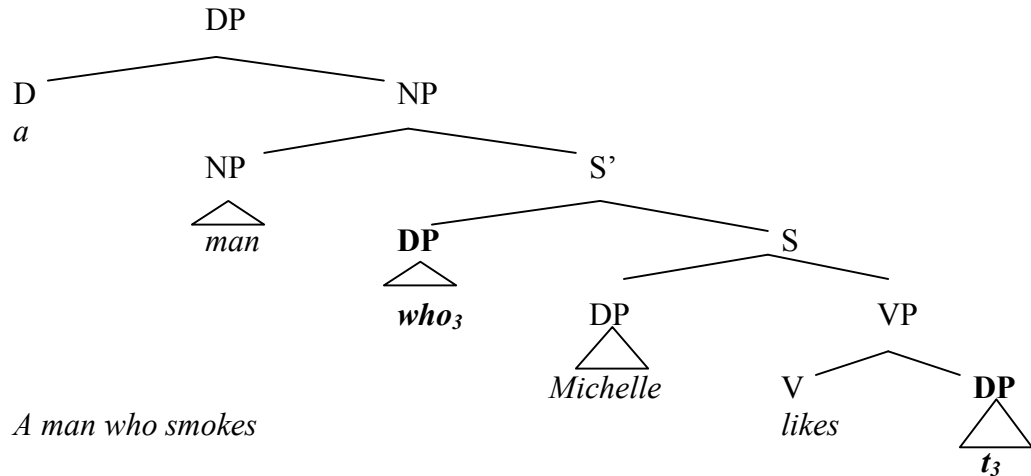
***The indices of the relative operator and its trace must be the same!***



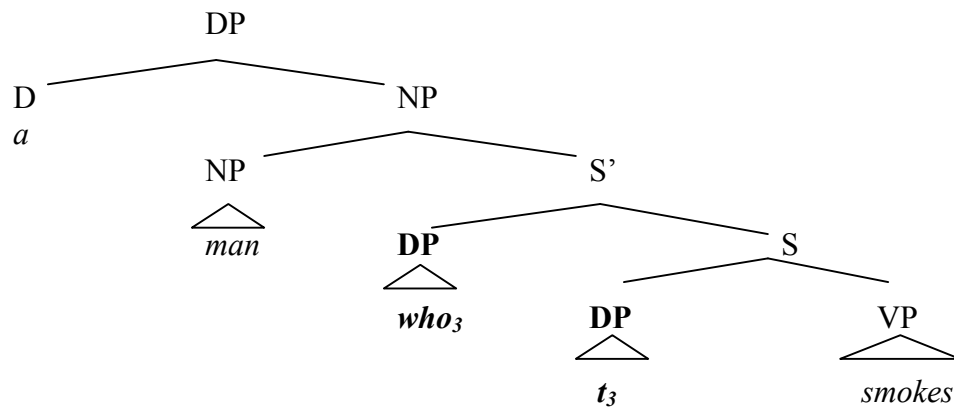
(8) **The Upshot**

The following is the syntax which we will assume for “a man who Michelle likes” and “a man who smokes”.

a. *A man who Michelle likes*



b. *A man who smokes*



**1.2 The T-Conditions of Sentences Containing Relative Clauses**

The following seem to be accurate T-conditional statements. Let's therefore seek to derive them.

(9) **Targeted T-Conditional Statements**

- a. "Obama is a man who smokes" is T *iff* Obama is a man and Obama smokes.
- b. "Obama is a man who M. likes" is T *iff* Obama is a man and M. likes Obama.

(10) **Some Initial Generalizations**

From (9), and our other background assumptions...

...we can deduce what the extensions of the complex, modified NPs must be...

a. Targeting 'man who smokes'

- (i)  $[[ \text{Obama is a man who smokes} ]]$  = T *iff* (by FA, NN, TN)
- (ii)  $[[ \text{is a man who smokes} ]](\text{Obama})$  = T *iff* (by FA, NN, TN, LC)
- (iii)  $[[ \text{man who smokes} ]](\text{Obama})$  = T *iff* (by (9a))
- (iv) Obama is a man and Obama smokes.

**CONCLUSION:**

$[[ \text{man who smokes} ]]$  =  $[ \lambda x : x \text{ is a man and } x \text{ smokes} ]$

b. Targeting 'man who Michelle likes'

- (i)  $[[ \text{Obama is a man who Michelle likes} ]]$  = T *iff* (by FA, NN, TN)
- (ii)  $[[ \text{is a man who Michelle likes} ]](\text{Obama})$  = T *iff* (by FA, NN, TN, LC)
- (iii)  $[[ \text{man who Michelle likes} ]](\text{Obama})$  = T *iff* (by (9b))
- (iv) Obama is a man and Michelle likes Obama.

**CONCLUSION:**

$[[ \text{man who Michelle likes} ]]$  =  $[ \lambda x : x \text{ is a man and Michelle likes } x ]$

---

2. **Reasoning Towards the Compositional Semantics of Relative Clauses**

Thus far, we've seen that if we can obtain the result in (11), we'll have a system that can derive the T-conditions of the sentences in (1)!

(11) **Our Targeted Results**

- a.  $[[ \text{man who smokes} ]]$  =  $[ \lambda x : x \text{ is a man and } x \text{ smokes} ]$
- b.  $[[ \text{man who Michelle likes} ]]$  =  $[ \lambda x : x \text{ is a man and Michelle likes } x ]$

...But how do we obtain the result in (11)?

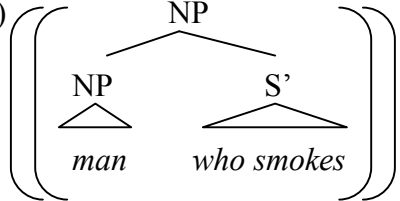
(12) **Key Observation**

If the equation in (12a) held, then we could derive the result in (11a) via the rule of PM.

a. Targeted Semantics of Relative Clause

$$[[who\ smokes]] = [\lambda x : x\ smokes]$$

b. Deriving (11a) via PM and (12a)

(i)  = (by PM)

(ii)  $[\lambda x : [[NP]](x) = T \text{ and } [[S']](x) = T]$  = (by NN, TN)

(iii)  $[\lambda x : [\lambda y : y \text{ is a man}](x) = T \text{ and } [[S']](x) = T]$  = (by notation)

(iv)  $[\lambda x : x \text{ is a man and } [[S']](x) = T]$  = (by (12a))

(v)  $[\lambda x : x \text{ is a man and } [\lambda y : y \text{ smokes}](x) = T]$  = (by notation)

(vi)  $[\lambda x : x \text{ is a man and } x \text{ smokes}]$

(13) **Working Hypotheses**

a. The type of a relative clause is <et>

b. The extension of a relative clause combines with the extension of the modified NP via PM.

...OK... but now how do we obtain the equation in (12a)?

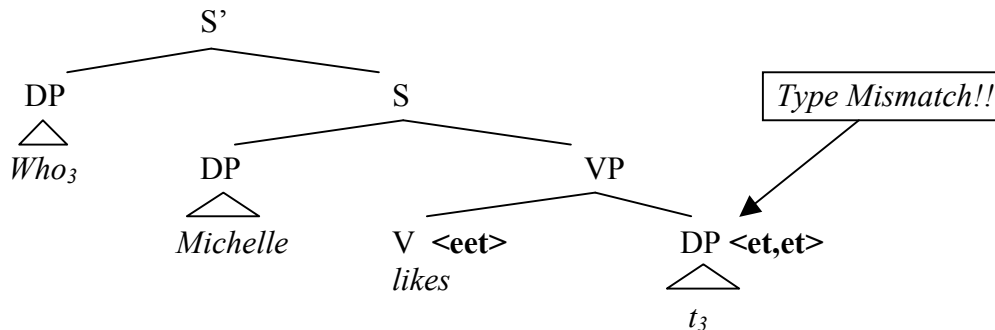
(14) **A Simple Idea**

Let's suppose that traces and relative pronouns are simply type <et,et> identity functions!

$$[[\text{who}_1 [\text{t}_1 \text{ smokes}]]] = [[\text{smokes}]] = [\lambda x : x \text{ smokes}]$$

(15) **Failure of the Simple Idea**

How, then, do we interpret relative clauses like that in (8a) / (11b)?



(16) **Immediate Question**

What *is* the extension of the relative clause in (15)?

a. Answer:

Following our 'working hypothesis' in (13), we should seek to obtain the following result:

$$[[ \text{who}_3 [ \text{Michelle} [ \text{likes } t_3 ] ] ] ] = [ \lambda x : \text{Michelle likes } x ]$$

b. Deriving (11b) via PM and (16a)

$$(i) \left( \left( \begin{array}{c} \text{NP} \\ \swarrow \quad \searrow \\ \text{NP} \quad \text{S}' \\ \swarrow \quad \searrow \\ \text{man} \quad \text{who Michelle likes} \end{array} \right) \right) = \text{(by PM)}$$

$$(ii) [ \lambda x : [[\text{NP}]](x) = T \text{ and } [[\text{S}']](x) = T ] = \text{(by NN, TN)}$$

$$(iii) [ \lambda x : [ \lambda y : y \text{ is a man } ](x) = T \text{ and } [[\text{S}']](x) = T ] = \text{(by notation)}$$

$$(iv) [ \lambda x : x \text{ is a man and } [[\text{S}']](x) = T ] = \text{(by (16a))}$$

$$(v) [ \lambda x : x \text{ is a man and } [ \lambda y : \text{Michelle likes } y ](x) = T ] = \text{(by notation)}$$

$$(vi) [ \lambda x : x \text{ is a man and Michelle likes } x ]$$

(17) **Summary: Targeted Equations**

a.  $[[ \text{who}_1 [ t_1 \text{ smokes } ] ]]$  =  $[ \lambda x : x \text{ smokes } ]$

b.  $[[ \text{who}_3 [ \text{Michelle} [ \text{likes } t_3 ] ] ]]$  =  $[ \lambda x : \text{Michelle likes } x ]$

*How can we derive these equations?...*

(18) **Crucial Observation**

Just as there are two syntactic ‘pieces’ to the English relative clause – the relative operator and the trace (gap) ...

*... there are two ‘pieces’ to the lambda expressions representing their extension – the lambda operator and the variable that it binds!...*

*...AND THESE TWO PIECES SEEM TO CORRESPOND WITH ONE ANOTHER!*

a.  $[[ \text{who}_1 [ t_1 \text{ smokes } ] ]]$   
 $\updownarrow$   
 $[[ \lambda x : x \text{ smokes } ]]$

b.  $[[ \text{who}_3 [ \text{Michelle} [ \text{likes } t_3 ] ] ]]$   
 $\updownarrow$   
 $[[ \lambda x : \text{Michelle likes } x ]]$

(19) **New Working Hypotheses**

- a. The type of a relative clause is  $\langle et \rangle$
- b. The extension of a relative clause combines with the extension of the modified NP via PM.
- c. **The relative operator is (more or less) interpreted as the lambda operator in the description of the  $\langle et \rangle$  function denoted by the relative clause.**
- d. **The trace (gap) is interpreted as the variable bound by the lambda operator in the description of the  $\langle et \rangle$  function denoted by the relative clause.**

*So, let's now work to formally implement the hypotheses in (19c) and (19d)...*

*... Doing so, however, will require us to introduce some new notation and a new semantic rule.*

### 3. Formally Implementing Our Hypothesis

#### (20) First Ingredient: Modified Variable Assignments

Let  $g$  be a variable assignment, let  $n$  be an index, and let  $a$  be some entity.

We will write ' $g (n / a)$ ' to mean 'the variable assignment which is exactly like  $g$ , except that it maps the index  $n$  to the entity  $a$ .

#### (21) Illustrating the New Notation

Let  $g$  be the following variable assignment: { <1 , Joe> , <2 , Obama> , <3 , Michelle> }

a.  $g (2 / Michelle) = \{ \langle 1 , Joe \rangle , \langle 2 , Michelle \rangle , \langle 3 , Michelle \rangle \}$

b.  $g (4 / Hillary) = \{ \langle 1 , Joe \rangle , \langle 2 , Obama \rangle , \langle 3 , Michelle \rangle , \langle 4 , Hillary \rangle \}$

c.  $g (3/Hillary) (1/Nancy) = \{ \langle 1 , Nancy \rangle , \langle 2 , Obama \rangle , \langle 3 , Hillary \rangle \}$

*The second ingredient in our formalization will be a new rule of semantic composition, specifically tied to relative operators....*

#### (22) The Rule of Predicate Abstraction (PA)

If  $X$  is a branching node whose daughters are  $Y$  and  $Z$ , and if  $Y$  is a relative pronoun bearing the index  $n$ , then **for any variable assignment  $g$** :

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

#### (23) Some Comments

- The rule in (22) implements our 'working hypothesis' in (19c): *the relative operator is interpreted (more or less) as the lambda operator in the description of the extension of the relative clause.*
- The rule in (22) states that the sister to the relative pronoun is interpreted relative to the *new* variable assignment ' $g (n / x)$ '.  
This is the variable assignment just like  $g$  except the index of the relative operator ( $n$ ) is mapped to the variable bound by the lambda operator ( $x$ ).
- Rules of the kind in (22) – which are tied to specific lexical items – are called *syncategorematic*.

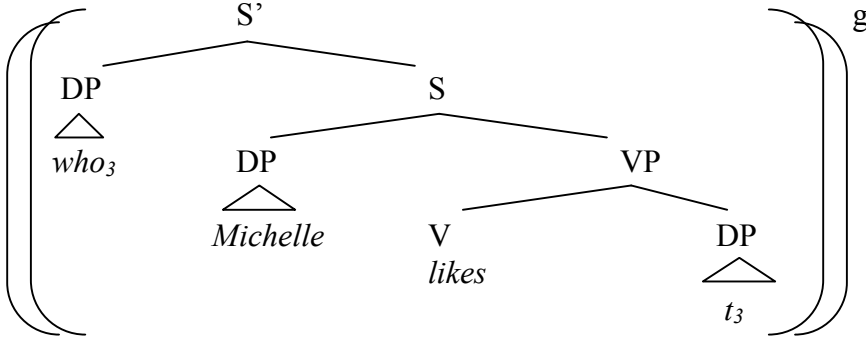
Finally, we need a rule that will allow us to interpret traces, and which is in line with our 'working hypothesis' in (19d)...

(23) **The Pronouns-and-Traces Rule (PR) [Heim & Kratzer (1998: 111)]**

If X is a pronoun *or a trace*, and g is a variable assignment, and n is an index in the domain of g, then  $[[X_n]]^g = g(n)$ .

*These three ingredients are enough to deliver our desired equations in (17)!*

(24) **Sample Derivation**

- a.  = (by PA)
- b.  $[\lambda x_e : [[S]]^{g(3/x)} = T]$  = (by FA, NN, TN)
- c.  $[\lambda x_e : [[VP]]^{g(3/x)}(\text{Michelle}) = T]$  = (by FA, NN)
- d.  $[\lambda x_e : [[likes]]^{g(3/x)} ([[t_3]]^{g(3/x)})(\text{Michelle}) = T]$  = (by TN)
- e.  $[\lambda x_e : [\lambda y_e : [\lambda z_e : z \text{ likes } y]] ([[t_3]]^{g(3/x)})(\text{Michelle}) = T]$  = (by PR)
- f.  $[\lambda x_e : [\lambda y_e : [\lambda z_e : z \text{ likes } y]] (\mathbf{g(3/x)}(\mathbf{3}))(\text{Michelle}) = T]$  = (by notation)
- g.  $[\lambda x_e : [\lambda y_e : [\lambda z_e : z \text{ likes } y]] (\mathbf{x})(\text{Michelle}) = T]$  = (by LC)
- h.  $[\lambda x_e : [\lambda z_e : z \text{ likes } x]](\text{Michelle}) = T]$  = (by notation)
- i.  $[\lambda x_e : \text{Michelle likes } x]$

The reader is invited to derive, via a similar proof, the equation in (17a).

#### 4. The Big Conclusion

Using the machinery introduced in the preceding sections, we can now compute the targeted T-conditions in (9), repeated below:

##### (25) Targeted T-Conditional Statements

- a. “Obama is a man who smokes” is T *iff* Obama is a man and Obama smokes.
- b. “Obama is a man who M. likes” is T *iff* Obama is a man and M. likes Obama.

The following derivation (which is greatly simplified) shows how this can now be done:

##### (26) Sample Derivation

*Let ‘g’ be any variable assignment:*

- a.  $[[ \text{Obama is a man who}_3 \text{ Michelle likes } t_3 ]]^g = T \text{ iff}$  (by FA, NN, TN)
- b.  $[[ \text{ is a man who Michelle likes } ]]^g (\text{Obama}) = T \text{ iff}$  (by FA, NN, TN, LC)
- c.  $[[ \text{ man who Michelle likes } ]]^g (\text{Obama}) = T \text{ iff}$  (by PM, NN, TN, LC)
- d.  $[ \lambda x : x \text{ is a man and } [[ \text{ who}_3 \text{ Michelle likes } t_3 ]]^g (x) = T ](\text{Obama}) = T \text{ iff}$   
(by PA, FA, NN, TN, PR)
- e.  $[ \lambda x : x \text{ is a man and } [ \lambda y_e : \text{Michelle likes } y ] (x) = T ](\text{Obama}) = T \text{ iff}$  (by notation)
- f.  $[ \lambda x : x \text{ is a man and Michelle likes } x ](\text{Obama}) = T \text{ iff}$  (by notation)
- g. Obama is a man and Michelle likes Obama.

*A similar derivation will allow us to derive the T-conditional statement in (25a)...*

---

#### 5. Pronouns and Binding

##### (27) Two Key Properties of Our Treatment of Relatives

- a. Traces are interpreted via the same rule as pronouns (PR), and are interpreted via the same ‘mechanism’ as pronouns (the variable assignment g).
- b. Relative pronouns are interpreted as ‘shifting’ or ‘manipulating’ the variable assignment g (the same mechanism as interprets pronouns).

(28) **Question: WHY?**

*Why posit this relationship between the semantics of traces (and relative operators) and the semantics of pronouns?*

(29) **Answer: Bound Pronouns**

This architecture allows us to capture the fact that pronouns sometimes seem to function semantically as bound variables, *just like traces*.

(30) **Example of Bound Pronoun**

- a. The man who loves the woman who loves **him** smokes.
- b. Two Possible Readings:
  - (i) *Context: Speaker points at Dave*  
The man who loves the woman who loves **Dave** smokes.
  - (ii) *Context: (no pointing)*  
The **x** such that **x** is a man and **x** loves the woman that loves **x** smokes.
- c. Terminology
  - (i) Reading (30bi) = ‘the referential reading of *him*’
  - (ii) Reading (30bii) = ‘the **bound** reading of *him*’

**Side-Note:**

The ‘bound reading’ of (30a) requires that there be only one man **x** such that **x** loves the woman that loves **x**...

However, the ‘referential reading’ of (30a) is consistent with there being *many* such men.

(31) **Key Fact**

- Our system is able to capture *both* these readings of (30).
- Each reading corresponds to a different syntax / indexing of the sentence’s DPs

a. Structure Assigned the ‘Referential Reading’

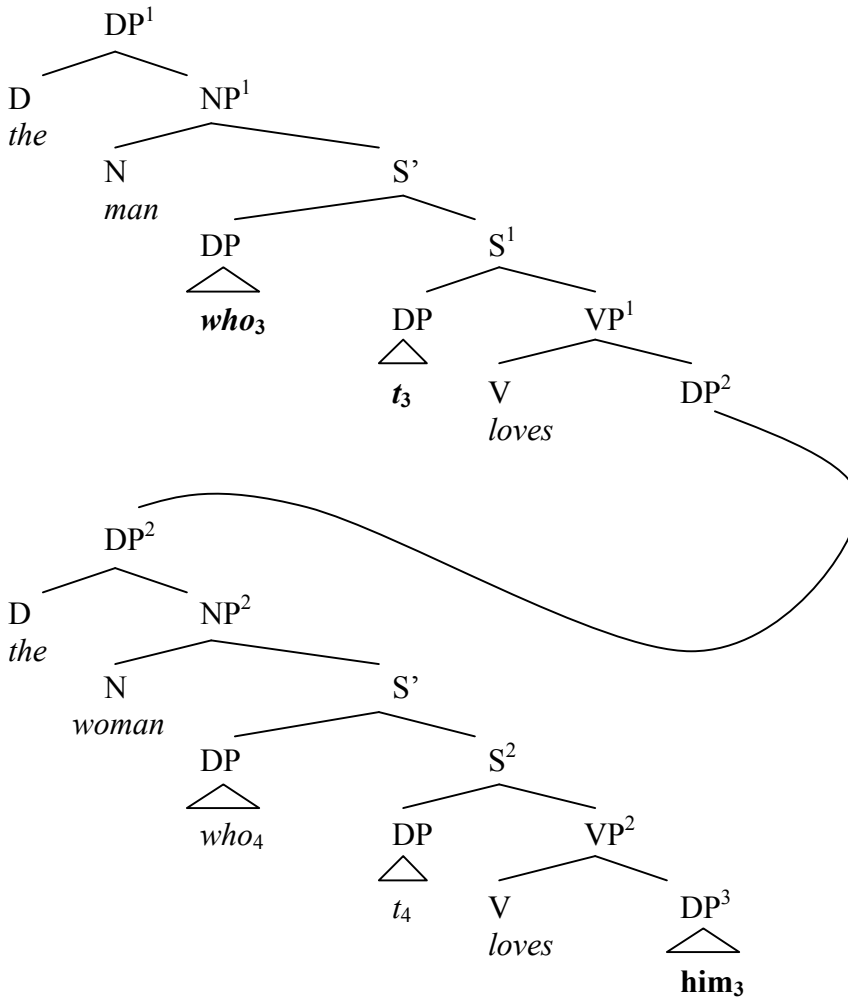
[ the [ man [ who<sub>3</sub> [ t<sub>3</sub> loves the woman [ who<sub>4</sub> [ t<sub>4</sub> loves **him**<sub>5</sub> ]]]]]] smokes.

b. Structure Assigned the ‘Bound Reading’

[ the [ man [ **who**<sub>3</sub> [ t<sub>3</sub> loves the woman [ who<sub>4</sub> [ t<sub>4</sub> loves **him**<sub>3</sub> ]]]]]] smokes.

(32) Derivation of the Bound Reading

a. Structure



b. Computation

Let 'g' be any variable assignment

- a.  $[[DP^1]]^g =$  (by FA)
- b.  $[[the]]^g ([[NP^1]]^g) =$  (by PM)
- c.  $[[the]]^g ([\lambda x: [[N]]^g(x) = T \ \& \ [[S']]^g(x) = T]) =$  (by TN)
- d.  $[[the]]^g ([\lambda x: x \text{ is a man} \ \& \ [[S']]^g(x) = T]) =$  (by TN, LC)
- e.  $[the \text{ unique } y \text{ such that } y \text{ is a man} \ \& \ [[S']]^g(y) = T] =$  (by PA)

- f. [the unique y such that y is a man &  $[\lambda x: [[S^1]]^{g(3/x)} = T](y) = T] =$  (by notation)
- g. [the unique y such that y is a man &  $[[S^1]]^{g(3/y)} = T] =$  (by FA, NN)
- h. [the unique y such that y is a man &  $[[VP^1]]^{g(3/y)}( [[t_3]]^{g(3/y)} ) = T] =$  (by PR)
- h. [the unique y such that y is a man &  $[[VP^1]]^{g(3/y)}( \mathbf{g(3/y) (3)} ) = T] =$
- i. [the unique y such that y is a man &  $[[VP^1]]^{g(3/y)}(\mathbf{y}) = T] =$  (by FA, NN, TN)
- k [the unique y such that y is a man &  
 $[\lambda x: \lambda z: z \text{ loves } x]( [[DP^2]]^{g(3/y)} )(\mathbf{y}) = T] =$  (by notation)
- l. the unique y such that y is a man & y loves  $[[DP^2]]^{g(3/y)} =$  (by FA)
- m. the unique y such that y is a man & y loves  $[[the]]^{g(3/y)}( [[NP^2]]^{g(3/y)} ) =$  (by PM, TM)
- n. the unique y such that y is a man & y loves  
 $[[the]]^{g(3/y)}([\lambda x: x \text{ is a woman \& } [[S'] ]^{g(3/y)}(x) = T]) =$  (by TN, LC)
- o. [the unique y such that y is a man & y loves  
the unique z such that z is a woman &  $[[S'] ]^{g(3/y)}(z) = T] =$  (by PA)
- p. [the unique y such that y is a man & y loves  
the unique z such that z is a woman  
&  $[\lambda x: [[S^2]]^{g(3/y)(4/x)} = T](z) = T] =$  (by notation)
- q. [the unique y such that y is a man & y loves  
the unique z such that z is a woman &  $[[S^2]]^{g(3/y)(4/z)} = T] =$  (by FA, NN, TN)
- r. [the unique y such that y is a man & y loves  
the unique z such that z is a woman &  
 $[\lambda x: \lambda s: s \text{ loves } x]( [[him_3]]^{g(3/y)(4/z)} ) ( [[t_4]]^{g(3/y)(4/z)} ) = T] =$  (by PR)
- s. [the unique y such that y is a man & y loves the unique z such that  
z is a woman &  $[\lambda x: \lambda s: s \text{ loves } x](\mathbf{y})(\mathbf{z}) = T] =$  (by notation)
- t. the unique y such that y is a man &  
y loves the unique z such that z is a woman and z loves y.

(33) **Summary**

Our system derives the ‘bound reading’ of (30a) if the following holds:

- The pronoun *him* receives the same index as the c-commanding relative operator.

Regardless of the assignment function *g*, such a structure will receive the ‘bound’ reading in (30bii)